



**本科毕业设计（论文）**

**基于嵌入式Linux系统的地铁AFC设备的AGM读写器设计与实现**

|  |  |
| --- | --- |
| **学 院** | **信息工程学院** |
| **专 业** | **通信工程** |
| **年级班别** | **2019级（3）班** |
| **学 号** | **3119002470** |
| **学生姓名** | **梁丞奂** |
| **指导教师** | **郝禄国** |

2022年11月

**基于嵌入式Linux 系统的地铁AFC 设备的AGM 读写器设计与实现 梁丞奂** **信息工程学院**

摘要

智能读写器是专为轨道交通自动售检票系统应用项目设计和生产的轨道交通行业应用智能式IC卡读写模块产品。其在轨道交通自动售检票系统中有着举足轻重的地位和作用。作为票卡交互的部件，其基本功能是对票卡的读、写处理，完成设备与票卡的读写接口。如今AFC系统标准读卡器应用日趋广泛，为了适应乘客的丰富需求，其性能以及功能的进一步完善与进步是大势所趋。

本人对地铁自动检票设备中自动检票闸机进行了相关研究，并通过研究嵌入式系统的基本原理，学习嵌入式系统应用软件开发技术，掌握基于Linux的嵌入式系统软件开发的基本流程和核心技术。进而设计开发了装配在自动检票闸机上的智能读写器。本论文综合调研了目前地铁的自动检票系统、国内对智能式读写器的研究现状以及现有读卡器的提升需求，从通信技术、硬件方案、软件协议等角度对读写器进行系统的阐述。并主要从软件设计方案详细介绍了本文的研究内容和设计开发成果，最后辅于轨道交通相关的测试软件对设计的智能读写器软件进行测试。读写器的测试结果表明其能够满足不同轨道运营商系统的应用及城市通卡运营商系统的应用、满足各种新型票卡的验票交易、满足自动检票闸机验票、开关闸等业务功能的报文命令交互。

**关键词：**嵌入式，Linux系统，自动检票闸机，智能读写器，串口通信

Abstract

Intelligent read-write device is an intelligent IC card read-write module product specially designed and produced for the rail transit automatic fare collection system application project. It plays an important role in the automatic fare collection system of rail transit. As a part of ticket card interaction, its basic function is to read and write the ticket card and complete the reading and writing interface between the equipment and the ticket card. Nowadays, the standard card reader of AFC system is increasingly widely used. In order to meet the rich needs of passengers, its performance and function are further improved and improved.

I have carried out relevant research on the automatic ticket check gate in the subway automatic ticket check equipment, and learned the embedded system application software development technology through studying the basic principle of the embedded system, and mastered the basic process and core technology of the embedded system software development based on Linux. Furthermore, an intelligent read-write device installed on the automatic gate is designed and developed. This paper comprehensively investigates the current automatic ticket checking system of the subway, the research status of intelligent readers and writers in China, and the improvement requirements of existing readers, and systematically expounds the readers and writers from the perspectives of communication technology, hardware solutions, software protocols, etc. It mainly introduces the research content and design and development results of this paper in detail from the software design scheme, and finally tests the designed intelligent reader software with the help of the relevant test software of rail transit. The test results of the reader-writer show that it can meet the application of different rail operator systems and the application of the urban pass card operator systems, meet the ticket checking transactions of various new types of ticket cards, and meet the message command interaction of the business functions such as the automatic ticket check machine ticket checking, switch and so on.

**Key words**：Embedded, Linux, Automatic Gate, Intelligent Reader and Writer, Serial Communication

**目录**

[摘要 3](#_Toc124451140)

[Abstract 4](#_Toc124451141)

[1 绪论 1](#_Toc124451142)

[1.1 选题背景及目的 1](#_Toc124451143)

[1.2 国内研究现状 2](#_Toc124451144)

[1.3 论文结构及内容详细安排 4](#_Toc124451145)

[2 方案设计与基本理论 5](#_Toc124451146)

[2.1 需求分析 5](#_Toc124451147)

[2.1.1 总体功能需求 5](#_Toc124451148)

[2.1.2 模块化功能需求 6](#_Toc124451149)

[2.2 基础理论和原理分析 7](#_Toc124451150)

[2.2.1 开发环境 7](#_Toc124451151)

[2.2.2 开发工具 10](#_Toc124451152)

[2.2.3 通信技术 11](#_Toc124451153)

[2.2.4 读写技术 15](#_Toc124451154)

[2.3 方案设计 17](#_Toc124451155)

[2.3.1 整体方案设计 17](#_Toc124451156)

[2.3.2 硬件方案整体设计 17](#_Toc124451157)

[2.3.3 软件方案整体设计 19](#_Toc124451158)

[2.3.4 软件方案模块设计 25](#_Toc124451159)

[2.3.5 方案设计标准 32](#_Toc124451160)

[2.3.6 方案总结 32](#_Toc124451161)

[3 概要设计 33](#_Toc124451162)

[3.1 读写器软件概要设计 33](#_Toc124451163)

[3.1.1 公用接口子模块设计 33](#_Toc124451164)

[3.1.2 AGM机业务功能模块介绍 38](#_Toc124451165)

[3.1.3 票卡读写模块介绍 40](#_Toc124451166)

[3.1.4运营参数模块介绍 40](#_Toc124451167)

[3.1.5蓝牙通信模块介绍 43](#_Toc124451168)

[3.2 数据库设计 44](#_Toc124451169)

[3.2.1 应用数据调用图 44](#_Toc124451170)

[3.2.2 数据模型设计 44](#_Toc124451171)

[4 详细设计 45](#_Toc124451172)

[**4.1 软件初始模块详细设计** 45](#_Toc124451173)

[**4.2公用接口模块详细设计** 45](#_Toc124451174)

[4.2.1串口通信编程 45](#_Toc124451175)

[4.2.2 设备初始化 55](#_Toc124451176)

[4.2.3 版本信息获取 59](#_Toc124451177)

[4.2.4 获取SAM卡信息 59](#_Toc124451178)

[4.2.5 设备降级模式 59](#_Toc124451179)

[4.2.6 获取票价 59](#_Toc124451180)

[4.2.7 参数管理 59](#_Toc124451181)

[4.2.8 时间同步 59](#_Toc124451182)

[4.2.9 获取寄存器值 59](#_Toc124451183)

[4.2.10 获取读卡器当前信息 60](#_Toc124451184)

[**4.3 AGM机业务功能模块详细设计** 60](#_Toc124451185)

[4.3.1通道类型处理 60](#_Toc124451186)

[4.3.2入闸处理 62](#_Toc124451187)

[4.3.3出闸处理 64](#_Toc124451188)

[5 系统测试 64](#_Toc124451189)

[5.1 测试环境 64](#_Toc124451190)

[5.2 功能测试 64](#_Toc124451191)

[总结 65](#_Toc124451192)

[参考文献 65](#_Toc124451193)

[致谢 67](#_Toc124451194)

# 绪论

## 选题背景及目的

集成电路卡 (Integrated Circuit Card，IC卡)，也称智能卡(Smart Card)、智慧卡(Intelligent Card)、微电路卡(Microcircuit Card)或微[芯片](https://baike.baidu.com/item/%E8%8A%AF%E7%89%87/32249?fromModule=lemma_inlink)卡等。它是将一个[微电子](https://baike.baidu.com/item/%E5%BE%AE%E7%94%B5%E5%AD%90/1410350?fromModule=lemma_inlink)[芯片](https://baike.baidu.com/item/%E8%8A%AF%E7%89%87/32249?fromModule=lemma_inlink)嵌入符合ISO7816标准的卡基中，做成卡片形式。IC卡与读写器之间的通讯方式可以是[接触](https://baike.baidu.com/item/%E6%8E%A5%E8%A7%A6/5692?fromModule=lemma_inlink)式也可以是非接触式。IC卡产品应用于金融、交通、医疗、身份证明等行业，它将微电子技术和计算机技术结合在一起，提高了人们工作、生活的现代化程度。电子科技进步一日千里，在短短几年内，IC卡技术在轨道交通自动售检票（AutomaticFareCollection）系统的应用由研究摸索迅速膨胀为大规模的实际应用，非接触式IC卡以其储存量较大，保密性较强，可实现一卡多个应用的特性逐步取代了磁卡的地位，如今已成为各城市轨道交通收费系统的首选票质媒介。而智能读写器是专为轨道交通自动售检票系统应用项目设计和生产的轨道交通行业应用智能式IC卡读写模块产品。其在轨道交通自动售检票（AFC）系统中有着举足轻重的地位和作用。作为票卡交互的部件，其基本功能是对票卡的读、写处理，完成设备与票卡的读写接口。

AFC系统标准读写器在功能上需实现以下功能：实现读写器与票卡之间的通信，主要包括对票卡数据的访问和读写功能；为票卡提供所需的工作能量；实现读写器与主控设备（上位机）之间的交互通信；实现票卡的读写；具备能够存储大量数据的功能（日志、交易数据、参数文件存储）；具有防冲突机制（寻卡多卡判定）。从软件应用层面，如今的读写器从传统读写器模式逐步发展为大读写器模式。两者的本质区别是大读写器中封装了完整的票卡业务处理程序，而传统读写器的票卡处理封装在AFC设备主控程序中。这些票卡业务包括：票种、票价表等各类票卡参数的使用；车票数据读取、分析；售、补、充、进出站等各类票卡业务处理逻辑；执行各类票卡业务并产生对应的交易数据。

智能式IC卡读写模块产品具备外形小巧、功能完善、便于安装等特点，可适用于自动售检票系统终端等城市轨道交通行业终端。如今AFC系统标准读卡器应用日趋广泛，为了适应乘客的丰富需求，其性能以及功能的进一步完善与进步是大势所趋。传统的AFC建设中技术以AFC集成厂商为主，各条线路各自招标，造成关键设备的读写器的供应商不同，硬件和软件不统一，基本不存在互换性。如今网络化建设和运营需求对关键核心设备的互换性要求提出了很高的要求：AFC智能读写器应按照线网级应用进行标准化设计和开发，票务处理软件朝着内置于读写器的方向发展，从根本上解决不同供货商设备的互通、互换，以及AFC设备适应不断发展的新票种、新业务、新应用的需求。因此，开发一种作为城市轨道交通AFC设备中的AGM机的高效智能式读卡器势在必行。

## 国内研究现状

我国城市轨道交通车站的自动售检票设备（AutomaticFareCollection），最初是来自外国，近年来我国已进行了大量的开发研制工作，提出了多种形式的产品，技术水平也在不断提高。国内轨道交通AFC系统的发展经历了从无到有的过程，随着计算机技术和软件的发展，我国城市轨道交通AFC的技术已与城市一卡通接轨，实现城市甚至城市区间的一卡通。

地铁作为现代城市的主要轨道交通工具，是大部分居民出行的优先选择。伴随着地铁的快速发展，自动售检票(AFC)设备的数量也成倍增长，运营效率极大提高的同时，日益增长的客流也对AFC设备提出了更加严峻的考验。AFC系统作为直接面向乘客的系统，能够完成自动售票、检票等基础服务。同时，AFC系统还能够进行票务管理、统计清分、财务结算等数据统计分析，为轨道交通科学合理地运营提供有力的数据支持。

基于通信、网络、计算机和自动控制等技术，自动售检票系统是一个高度自动化的系统，实现了全过程的自动化系统，包括：售检票、计费、收费、清分统计（指当日交易明细的汇总及清算净额）和管理等功能。城市轨道交通自动售检票系统的基本架构一般分为五个层次，就是通常所说的五层架构，分别为清分中心、线路中央计算机层、车站计算机层、车站终端设备层和车票五个层次。其中与乘客直接接触的主要是车票与车站终端设备，车站终端设备主要包括检票闸机、自动加值机、自动售票机与半自动售票机等，这些设备都与读卡器关系密切，尤其是AGM机即自动检票闸机。

自动检票闸机是公共场合不仅起到疏散人群的作用，同时还具备检票验票的功能。自动检票闸机是一种通道阻挡装置(通道管理设备)，用于管理人流并规范行人出入，自动检票闸机主要应用于地铁闸机系统、收费检票闸机系统、景区检票系统等、其最基本最核心的功能是实现一次只通过一人，可用于各种收费、门禁场合的入口通道处。自动检票机是闸机结合票务系统的一种具体应用。《城市轨道交通自动售检票系统技术条件》(GB／T20907--2007)规定，门式检票机在无回收票情况下通过能力不小于30AJmin。因此，理论上每台自动检票机高峰小时要通过1800位乘客。

在轨道AFC系统中，AGM机即检票闸机的功能是承担检票服务，位置往往设置在站厅付费区与非付费区之间。当乘客持单程票或储值票从非付费区进入付费区，通过闸机时，闸机上的智能读卡器就会读取车票的信息并写入入站时间、车站编号等信息，若车票满足相关规定，则给予放行。当乘客离开付费区时，闸机上的智能读卡器会再次读取车票的信息并检验其是否有效。对于有效单程票，回收后放行；对于储值票，扣除此次乘车费用后放行。往往在早晚高峰客流时，闸机的通行能力的其中一项主要影响因素就是智能读卡器的响应速率，对于市中心、客运站等节点处，高峰客流的通过需求与闸机的通过能力的矛盾更加尖锐。 

检票机是实现乘客自助进出站检票交易(在非付费区和付费区之间通行)的设备，凡持有有效车票的乘客通过检票机，其通道阻挡会自动解除(释放转杆或门扇开启)，允许乘客进出站。检票机按不同的功能可划分为进站检票机、出站检票机和双向检票机；按阻挡方式可划分为门式检票机和三杆式检票机：按通道宽带可划分普通检票机和宽道检票机(主要用于轮椅通过)。

## 论文结构及内容详细安排

本文的主要内容是基于嵌入式Linux系统的地铁AFC设备的AGM读写器的设计与实现。本文的章节层次及内容安排如下所示：

1、绪论，主要是通过介绍本课题的选题背景，并分析本课题的国内研究现状。以来设定好本论文的框架结构。

2、方案设计与基本理论，首先对本课题的需求进行分析，主要针对目前AFC设备对读写器的需求和读写器功能的需要等进行概述。其次分析本文读卡器的设计原理和基本理论。然后得出相关技术选型的方案。最后对读写器设计方案进行概述和设计标准的总结。

3、概要设计，首先划分本课题读写器的各个功能模块并画出其运行逻辑图。其次将大的功能模块划分为小的功能模块并画出其调用关系图。然后设计好数据库并画出数据库和功能模块之间的调用关系。最后对概要设计进行总结。

4、详细设计，首先对读卡器的具体业务逻辑进行具体说明并画出流程图。然后对实现该功能的代码模块进行详细说明。最后对详细设计进行总结

5、测试实现，采用地铁AFC设备中专门的测试软件对读写器各项功能进行测试并对测试结果进行分析和说明。

6、论文总结，对本课题进行全面细致的总结，指出制作本课题的需要改进的地方以及在本课题的收获。

# 方案设计与基本理论

## 需求分析

### 总体功能需求

轨道交通系统专业很多，但AFC系统的重要性不言而喻，它为乘客提供进展自动检票和出站自动售票的功能，同时也为轨道交通运营公司提供有效可靠的数据，作为科学管理的数据支持。读写器作为轨道交通AFC系统设备闸机的主要组成部分，需票具有高效、可靠、灵敏的响应能力，并须具有良好稳定的性能。即是电子收费终端部分中的智能读卡设备。智能式读写器是专为轨道交通自动售检票系统应用项目设计和生产的轨道交通行业应用智能式IC卡读写模块。该产品设计具备外形小巧、功能完善、便于安装等特点，可适用于自动售检票系统终端等城市轨道交通行业终端，也可被应用于金融IC卡接入或城市一卡通等领域。读卡器的设计应满足各种相关的技术标准，符合国际标准和国家标准。

读写器采用模块化结构设计，严格遵守轨道交通AFC标准进行设计，对外接口统一，设计的读写器满足以下需求：

**1、满足不同轨道运营商系统的应用及城市通卡运营商系统的应用。**

读写器底层预留大容量存储器，可满足现有和将来不同的应用需求，方便扩展。我方读写器是一个功能独立的、具备在安全认证模块的配合下，在读写范围内对车票完成车票分析和车票交易处理的整体功能统一的软、硬件综合体，无需外围其他设备参与读写、密钥计算功能，整个流程统一，对外接口一致。

读写器能独立完成所有与业务无关的票卡应用处理：非接触式IC卡检测、IC卡内的信息读取和写入、防冲突、选择IC卡应用、票卡密钥计算、密钥双向认证、票卡逻辑数据恢复、交易验证码产生、信用管理和交易流水号管理。

**2、满足地铁AFC设备AGM机的接入和命令报文的处理**

读写器能独立完成交易的全部处理过程，独立获取读写器运行所需要资源，与上位机程序仅使用简单接口（典型应用串口）互联，读写器与上位机仅有硬件连接，没有中间层软件。

**3、满足系统引入新功能和新票种（或新卡型）的要求**

读写器预留有新功能接口，预留手机支付功能接口，承诺满足系统引入手机支付功能和新票种（或新卡型）的要求，并可适应票卡数据结构的升级变化和跨行业的应用。

**4、满足连接智能装备并进行功能交互的要求**

智能设备是传统电气设备与计算机技术、数据处理技术、控制理论、传感器技术、网络通信技术、电力电子技术等相结合的产物。读卡器及核心模块可通过USB或串口通讯的方式与外围的智能设备进行连接,例如人脸识别、智能运维等以适合不同用户的需求。

### 模块化功能需求

**1、读卡器独立工作功能需求**

读卡器拥有强大的联网和脱机工作能力：当管理中心的监控电脑关机或通讯线路故障时，控制器能够正常地脱机工作，不会受到任何影响，在联网模式下，由管理层判断卡片是否合法，并触发例如开门动作的业务。搭载读卡器的AFC系统软件界面可弹出持卡人的照片、余额、次数等信息。

**2、接入终端设备功能需求**

获取外部模块传递的数据，从而完成外设的交互功能。能够设置设备标识，查询并识别设备，安全连接设备，检测通信状态，获取设备数据，对数据进行分析处理，安全断开设备，保留数据通信日志等。此需求要求读卡器有专门的串口设置。

**3、读卡器基本票务处理功能需求**

读写器能根据AGM设备的不同要求满足票卡的进出站、更新、查验等应用操作要求。读卡器用于实现对票卡的读写，在AFC系统的AGM机中大量使用。检票机读写器应能完成票卡的进出站检票交易功能。读卡器要满足将票卡处理的业务流程部署于读卡器内部，票卡业务的操作流程由读卡器独立完成，终端设备的工控机不涉及票卡业务的具体实施流程，工控机和读卡器之间只涉及操作命令、信息反馈等信息交互。

## 基础理论和原理分析

### 开发环境

[嵌入式](https://so.csdn.net/so/search?q=%E5%B5%8C%E5%85%A5%E5%BC%8F&spm=1001.2101.3001.7020" \t "_blank)开发就是指在[嵌入式操作系统](https://baike.baidu.com/item/%E5%B5%8C%E5%85%A5%E5%BC%8F%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/361747)下进行开发，包括在系统化设计指导下的硬件和软件以及综合研发。除暂且分离硬件的[EDA](https://baike.baidu.com/item/EDA/182009)研发以外，侧重的就是在一定硬件条件下的系统化设计和软件研发。

**1、嵌入式开发定义：**

嵌入式开发是指利用分立元件或集成器件进行电路设计、结构设计，再进行软件编程（通常是高级语言），实验，经过多轮修改设计、制作，最终完成整个系统的开发。嵌入式技术不是单纯的软件技术，也不是单纯的硬件技术，是一门如何在一个特定的硬件环境上开发与构建特定的可编程软件系统的综合技术。

**2、嵌入式开发流程：**

当前，嵌入式开发已经逐步规范化，在遵循一般工程开发流程的基础上，嵌入式开发有其自身的一些特点。主要包括系统需求分析（要求有严格规范的技术要求）、体系结构设计、软硬件及机械系统设计、系统集成、系统测试，最终得到产品。

（1）系统需求分析。确定设计任务和设计目标，并提炼出设计规格说明书，作为正式设计指导和验收的标准。系统的需求一般分功能性需求和[非功能性需求](https://baike.baidu.com/item/%E9%9D%9E%E5%8A%9F%E8%83%BD%E6%80%A7%E9%9C%80%E6%B1%82/8403760)两方面。功能性需求是系统的基本功能，如输入输出信号、操作方式等；非功能性需求包括系统性能、成本、功耗、体积、重量等因素。

（2）体系结构设计。描述系统如何实现所述的功能和非功能性需求，包括对硬件、软件和执行装置的功能划分，以及系统的软件、硬件选型等。一个好的体系结构是设计成功与否的关键。

（3）硬件/软件协同设计。基于体系结构，对系统的软件、硬件进行详细设计。为了缩短产品开发周期，设计往往是并行的。嵌入式系统设计的工作大部分都集中在软件设计上。

（4）系统集成。把系统的软件、硬件和执行装置集成在一起，进行调试，发现并改进单元设计过程中的错误。

（5）系统测试。对设计好的系统进行测试，看其是否满足规格说明书中给定的功能要求。

### 开发工具

**1、C++**

C语言是结构化和模块化的语言，适合处理较小规模的程序。对于复杂的问题，规模较大的程序，需要高度的抽象和建模时，C语言则不合适。为了解决软件危机，20世纪80年代，计算机界提出了OOP(objectorientedprogramming：面向对象)思想，支持面向对象的程序设计语言应运而生。1982年，BjarneStroustrup博士在C语言的基础上引入并扩充了面向对象的概念，发明了一种新的程序语言。为了表达该语言与C语言的渊源关系，命名为C++。因此：C++是基于C语言而产生的，它既可以进行C语言的过程化程序设计，又可以进行以抽象数据类型为特点的基于对象的程序设计，还可以进行面向对象的程序设计。C++语言的程序开发环境，为了方便测试，将调试环境做成了解释型。即开发过程中，以解释型的逐条语句执行方式来进行调试，以编译型的脱离开发环境而启动运行的方式来生成程序最终的执行代码。

开发C++应用程序，需要经过编写源程序、编译、连接程序生成可执行程序、运行程序四个步骤。生成程序是指将源码（C++语句）转换成一个可以运行的应用程序的过程。如果程序编写正确，那么通常只需按一个功能键，即可完成该过程。

第一步对程序进行编译，这需要用到编译器（compiler）。编译器将C++语句转换成机器码（也称为目标码）；如果该步骤成功执行，下一步就是对程序进行链接，这需要用到链接器（linker）。链接器将编译获得机器码与C++库中的代码进行合并。C++库包含了执行某些常见任务的函数（“函数”是子程序的另一种称呼）。例如，一个C++库中包含标准的平方根函数sqrt，所以不必亲自计算平方根。C++库中还包含一些子程序，它们把数据发送到显示器，并知道如何读写硬盘上的数据文件。

2、ARM平台

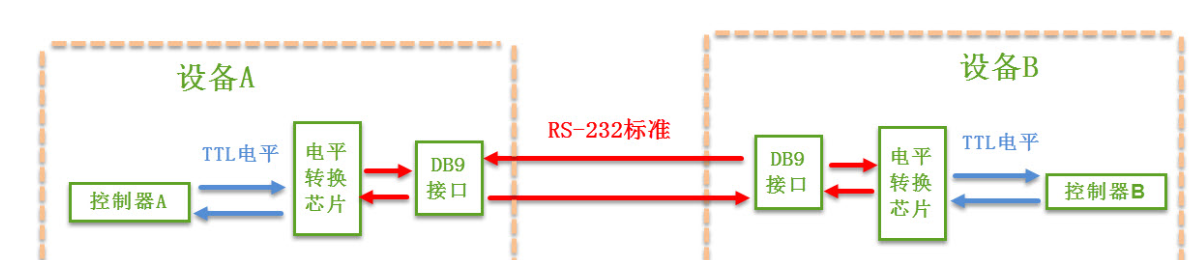
### 通信技术

**1、串口通信简介**：

串口通讯(SerialCommunication)是一种设备间非常常用的串行通讯方式，因为它简单便捷，因此大部分电子设备都支持该通讯方式，电子工程师在调试设备时也经常使用该通讯方式输出调试信息。

在计算机科学里，大部分复杂的问题都可以通过分层来简化。如芯片被分为内核层和片上外设；STM32标准库则是在[寄存器](https://so.csdn.net/so/search?q=%E5%AF%84%E5%AD%98%E5%99%A8&spm=1001.2101.3001.7020" \t "_blank)与用户代码之间的软件层。对于通讯协议，我们也以分层的方式来理解，最基本的是把它分为物理层和协议层。物理层规定通讯系统中具有机械、电子功能部分的特性，确保原始数据在物理媒体的传输。协议层主要规定通讯逻辑，统一收发双方的数据打包、解包标准。简单来说物理层规定我们用嘴巴还是用肢体来交流，协议层则规定我们用中文还是英文来交流。

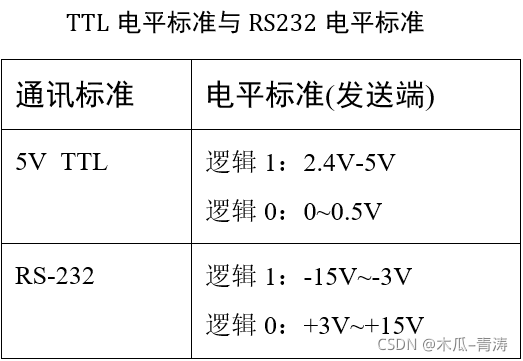
RS-232标准接口（又称EIARS-232）是常用的串行通信接口标准之一



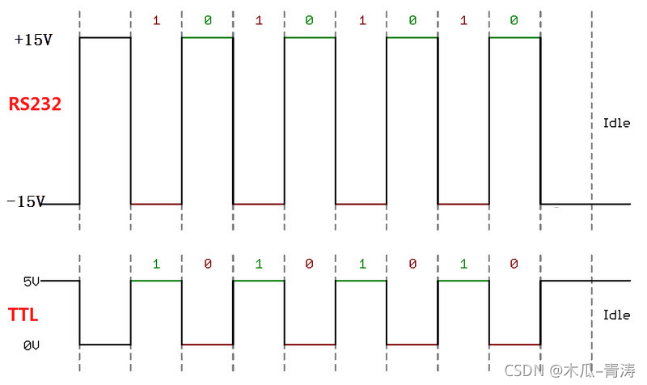
**2、电平标准**

在上面的通讯方式中，两个通讯设备的“DB9接口”之间通过串口信号线建立起连接，串口信号线中使用“RS-232标准”传输数据信号。由于RS-232电平标准的信号不能直接被控制器直接识别，所以这些信号会经过一个“电平转换芯片”转换成控制器能识别的“TTL标准”的电平信号，才能实现通讯。

根据通讯使用的电平标准不同，串口通讯可分为TTL标准及RS-232标准，TTL电平标准与RS232电平标准。



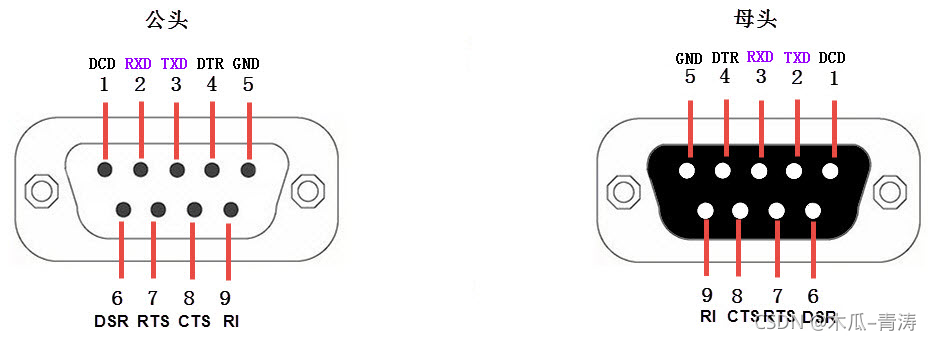
电子电路中常使用TTL的电平标准，理想状态下，使用5V表示二进制逻辑1，使用0V表示逻辑0；而为了增加串口通讯的远距离传输及抗干扰能力，它使用-15V表示逻辑1，+15V表示逻辑0。使用RS232与TTL电平校准表示同一个信号时的对比见图RS-232与TTL电平标准下表示同一个信号。

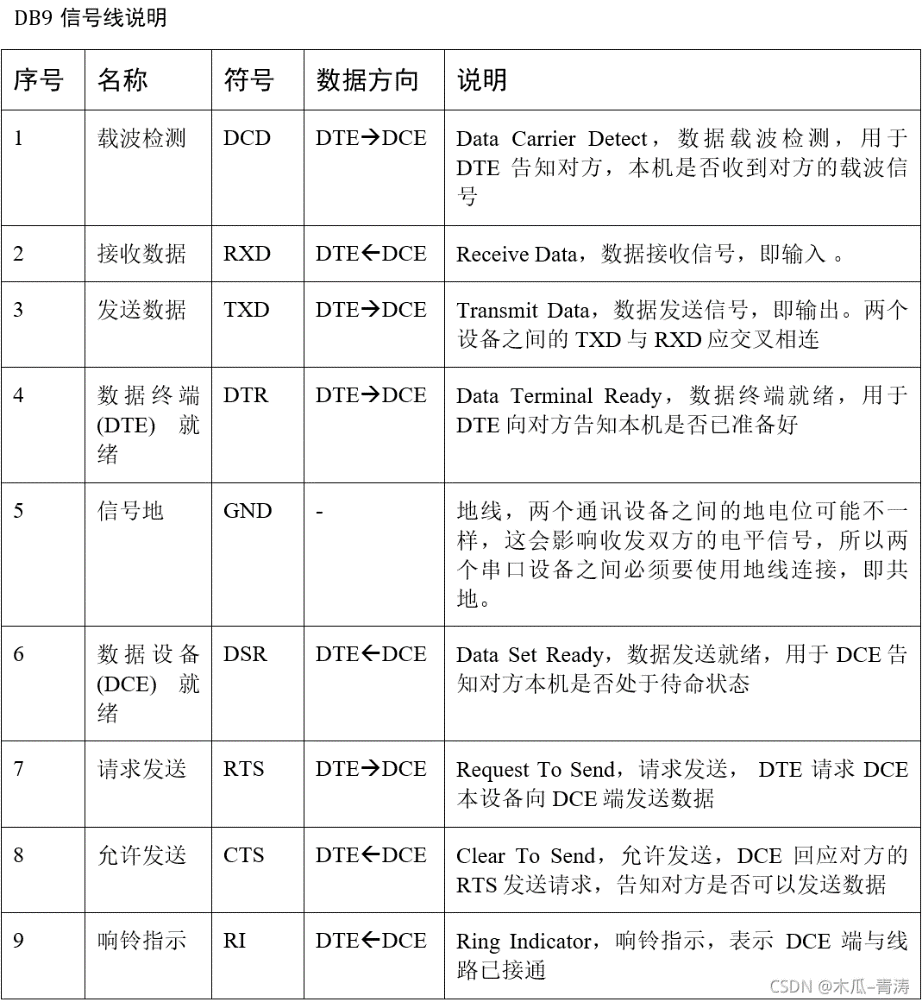


**3、信号线**

RS-232串口标准常用于计算机、路由与调制调解器(MODEN，俗称“猫”)之间的通讯，在这种通讯系统中，设备被分为数据终端设备DTE(计算机、路由)和数据通讯设备DCE(调制调解器)。我们以这种通讯模型讲解它们的信号线连接方式及各个信号线的作用。在旧式的台式计算机中一般会有RS-232标准的COM口(也称DB9接口)。



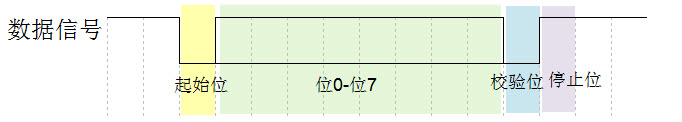




**4、协议层**

串口通讯的数据包由发送设备通过自身的TXD接口传输到接收设备的RXD接口。在串口通讯的协议层中，规定了数据包的内容，它由启始位、主体数据、校验位以及停止位组成，通讯双方的数据包格式要约定一致才能正常收发数据，其组成见图串口数据包的基本组成。

串口数据包的基本组成



（1）波特率：数据信号对载波的调制速率，串口异步通讯中由于没有时钟信号(如前面讲解的DB9接口中是没有时钟信号的)，所以两个通讯设备之间需要约定好波特率，即每个码元的长度，以便对信号进行解码，图串口数据包的基本组成中用虚线分开的每一格就是代表一个码元。常见的波特率为4800、9600、115200等。

（2）通讯的起始和停止信号：串口通讯的一个数据包从起始信号开始，直到停止信号结束。数据包的起始信号由一个逻辑0的数据位表示，而数据包的停止信号可由0.5、1、1.5或2个逻辑1的数据位表示，只要双方约定一致即可。

（3）有效数据：在数据包的起始位之后紧接着的就是要传输的主体数据内容，也称为有效数据，有效数据的长度常被约定为5、6、7或8位长。

（4）数据校验：在有效数据之后，有一个可选的数据校验位。由于数据通信相对更容易受到外部干扰导致传输数据出现偏差，可以在传输过程加上校验位来解决这个问题。校验方法有奇校验(odd)、偶校验(even)、0校验(space)、1校验(mark)以及无校验(noparity)。  
奇校验要求有效数据和校验位中“1”的个数为奇数，比如一个8位长的有效数据为：01101001，此时总共有4个“1”，为达到奇校验效果，校验位为“1”，最后传输的数据将是8位的有效数据加上1位的校验位总共9位。偶校验与奇校验要求刚好相反，要求帧数据和校验位中“1”的个数为偶数，比如数据帧：11001010，此时数据帧“1”的个数为4个，所以偶校验位为“0”。0校验是不管有效数据中的内容是什么，校验位总为“0”，1校验是校验位总为“1”。

### 读写技术

**1、NFC简介**

近场通信（NearFieldCommunication，简称NFC），是一种新兴的技术，使用了NFC技术的设备（例如移动电话）可以在彼此靠近的情况下进行数据交换，是由非接触式[射频](https://so.csdn.net/so/search?q=%E5%B0%84%E9%A2%91&spm=1001.2101.3001.7020" \t "_blank)识别（[RFID](https://baike.baidu.com/item/RFID/497249)）及互连互通技术整合演变而来的，通过在单一芯片上集成感应式读卡器、感应式卡片和点对点通信的功能，利用移动终端实现移动支付、电子票务、门禁、移动身份识别、防伪等应用。NFC可以理解为是RFID的一个子集。目前手机比较火的“碰一碰”功能就依托于该技术。NFC只是提供了一种简单、触控式的解决方案，可以让客户直观地和智能设备交换信息、访问所需内容、提供相应的服务。NFC技术支持设备之间进行非接触式点对点的数据传输，可以在10cm距离内交换数据，其传输速度主要有106Kbit/秒、212Kbit/秒或者424Kbit/秒三种。

**2、NFC工作模式**

**（1）**点对点模式(P2Pmode)和红外线差不多，用于数据交换，该模式传输距离较短，传输速度较快，功耗低（和蓝牙也类似）。将两个具备NFC功能的智能设备配对连接，能实现数据点对点传输，典型的例子比如智能手机自带的NFC可以让两台手机之间快速传输文件。

**（2）**有卡模式(Cardemulation)相当于一张采用RFID技术的IC卡。可以替代大量的IC卡（包括信用卡）、商场刷卡、交通卡、门禁，火车票等等。有卡模式有一个很大的优势就是卡片通过非接触读卡器的RF域进行供电，即便是寄主设备没电也可以工作。

**（3）**读卡器模式(Reader/writermode)其实就是通过支持NFC的智能手机或其它电子设备从带有NFC芯片的标签、贴纸、名片等媒介中读写信息。通常NFC标签是不需要额外外部供电的。当支持NFC的外设向NFC读写数据时，它会发送某种磁场，而这个磁场会自动向NFC标签供电。

**3、卡片读写标准**

[ISO/IEC14443](https://baike.baidu.com/item/iso14443/1967008)定义了TypeA、TypeB两种卡片读写标准：TYPEA卡在读写机上向卡传送信号时，是通过13.56MHz的射频载波传送信号。其采用方案为同步、改进的Miller编码方式，通过100%ASK传送；当卡向读写机具传送信号时，通过调制载波传送信号。使用847kHz的副载波传送Manchester编码。而TYPEB卡在读写机具向卡传送信号时，也是通过13.56MHz的射频载波信号，但采用的是异步、NRZ编码方式，通过用10%ASK传送的方案；在卡向读写机具传送信号时，则是采用的BPSK编码进行调制。对比两种卡型，可以看出，TypeB型与TypeA型卡相比有以下优势：

（1）芯片具有更高的安全性。接收信号时，不会因为能量损失而使芯片内部逻辑及软件工作停止。

（2）支持更高的通讯速率。TypeA最大的数据通讯速率为150Kbit/s-200Kbit/s，应用10%ASK技术的TypeB至少可支持400Kbit/s的速率。

（3）外围电路设计简单。读写机具到卡以及卡到读写机具的编码方式均采用NRZ方案，电路设计对称，设计时可使用简单的UARTS。

（4）抗干扰能力强。负载波采用BPSK调制技术，较TypeA方案降低了6dB的信号声。

在ISO/IEC14443-3规定了TYPEA，TYPEB的防冲突机制。二者防冲突机制的原理完全不同。前者是基于BIT冲突检测协议，后者则是通过字节、帧及命令完成防冲突。防冲突机制使非接触式IC卡能进行并行操作，及在多张卡同时进入有效操作区后，可对其进行有条不紊的操作，这样就使选定卡片的数据不受其它卡数据干扰，携带有多种卡的用户可不必寻找正确的一张卡，只用算法编程，读写机具即可自动做到选取正确的一张卡进行后续操作。这样方便了操作，提高了应用的并行性，也提高了系统的速度。

## 方案设计

### 整体方案设计

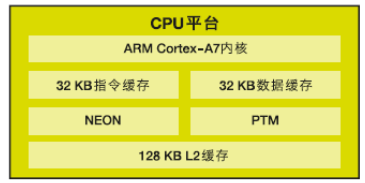
整体设计思想：根据需求分析，应用于轨道交通AFC系统的专用读卡器对体积、功耗和系统的可靠性均有较为严格的要求，而嵌入式系统正好可以满足AFC系统专用读卡器的这些要求。对嵌入式系统的定义是：以应用为中心、以计算机技术为基础、软硬件可裁剪、适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统。

从嵌入式系统的定义可以明显地得出其主要优点符合读卡器的设计要求，所以采用嵌入式平台对AFC系统专用读卡器进行设计。嵌入式系统具有嵌入性、系统性和智能性，具体来说嵌入式系统是嵌入到特定对象体系中的实体，是一个计算机系统同时具有嵌入式软硬件并能智能地实现预定的功能。专用性、实时性是嵌入式系统的主要特点的组成部分，专用性主要体现在嵌入式的专用处理器上，根据能、实现的功能不同有不同的专用处理器，如视频处理系统、语言处理系统等；实时性是因为嵌入式系统常被用作控制或者辅助操作设备的装置，因此要求其具有很好的实时性来满足对象系统的要求；除此之外，嵌入式系统还具有系统设备电子化、编程语言低级化、系统性能可测化以及开发系统专用化等特点。在嵌入式系统中，系统的控制核心是处理器，应具备良好的性能。

### 硬件方案设计

**1、读卡器芯片方案**

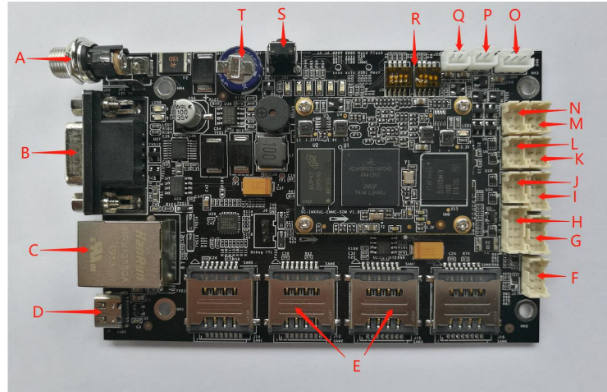
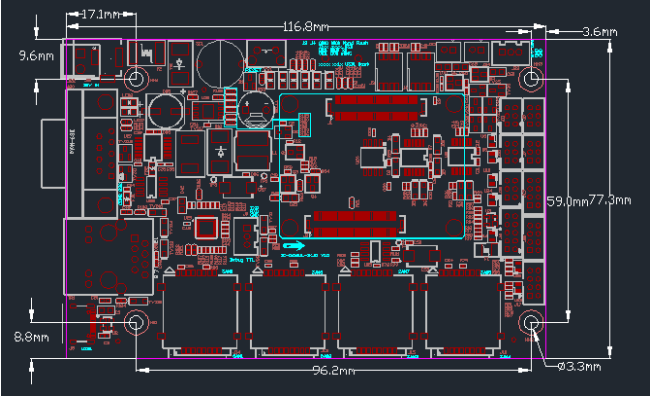
主控电路板选用ARMCortex-A7为核心CPU的MX6UL-C微控制器。ARMCORTEX-A7是一款经典的工业级CPU，其优越稳定的性能、高性价比，调试简便，外设接口丰富等特征使得该CPU被广泛地应用于各行各业。



由于ARMCortex-A7有着良好的性价比，其功能及性能可满足绝大多数的应用场合，生产工艺要求相对简单。该CPU有着巨大成熟的应用市场，普及程度高，有旺盛的生命力，可保证10年以上的供货期。根据的需求分析，本方案选用MX6UL-C作为读写器的主控制器。

**2、读卡器PCB方案**





A：电源接口

B：DB9 通讯口

C：RJ45 网络接口

D：USB OTG 接口

E：8 个 SAM 卡槽

F、G、I、H：天线接口

J、K、L：串口 TTL

N：串口 RS232(/devttymxc4、/dev/ttymxc5)

R：拨码开关

S：复位按键

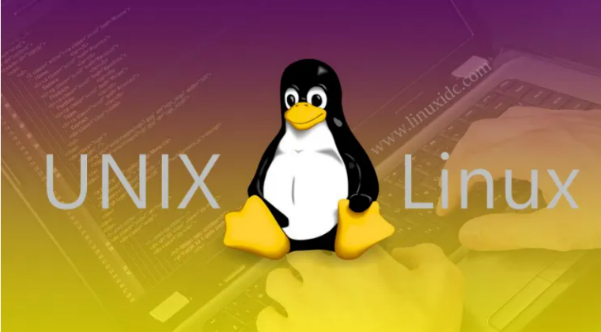
**3、读卡器天线方案**

读卡器选用外置天线，通过精心设置天线的形状和大小，使之在符合安装要求有着良好的性能，选择天线合适的品质因数Q，精确实现天线的阻抗匹包括线网系统中正在使用的票卡和在建线路采购的票卡品质因数。天线采用特征阻抗为50Ω的同轴电缆与读卡器相连系统专用读卡器的设计与实现不通过处理很难得到理想的电磁频率，所以读为了使EMC标EMC的方法是设是高频巴伦变压器，要连实现平衡到不平衡转换。使之在符合安装要求精确实现天线的阻抗匹包括线网系统中正在使用的票卡和在建线路采购的票卡）的同轴电缆与读卡器相连

### 软件方案设计

**1、读卡器软件架构**

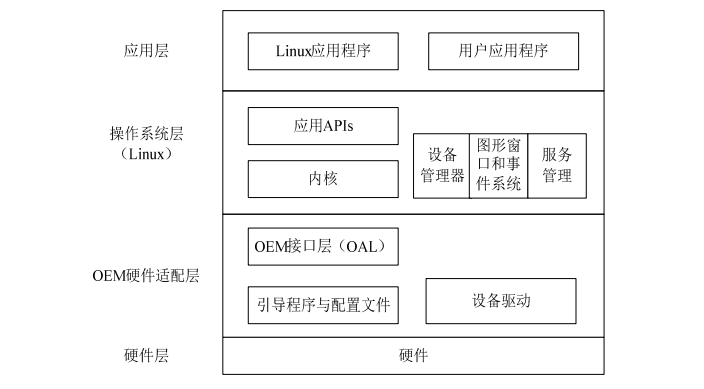
读卡器的功能要求比较复杂，需要多任务处理，且对系统的安全性和稳定性要求较高。由于嵌入式操作系统拥有完整的编程规范和统一的接口函数，可使用高级程序语言，从而使应用软件具有良好的代码可重用性和可移植性；同时，嵌入式操作系统具有多任务处理功能，能快速响应外部中断任务，能够显著地提高读卡器系统的整体性能。在选择读卡器操作系统时，主要考虑如下因素：(1)选用主流的、成熟的操作系统，该操作系统需拥有强盛的生命力，已经被广泛接受，将在可预见的未来，仍为主流操作系统。(2)应用软件可实现与城市轨道交通不同线路之间的兼容。(3)尽可能选用免费或低成本的系统，便于未来的大规模生产发布。



嵌入式Linux是将日益流行的Linux操作系统进行裁剪修改，使之能在嵌入式计算机系统上运行的一种操作系统。因此嵌入式Linux具备了Linux系统开源的优点，也具备了嵌入式这一特征。更重要的原因之一是其为免费系统，而且具有优异的性能，因其为开源代码的特点，软件移植起来相对容易，且应用软件开发周期短，具备良好的安全性、实时性、稳定性。由于市场发展前景广阔，Montavista、Lineo、Emi等专业公司开发了许多基于嵌入式Linux的专用产品，有行业协会如EmbeddedLinuxConsortum等，而IBM、Motorola、Intel等知名软件公司也纷纷支持，还有Internet上的大量嵌入式Linux爱好者的支持。嵌入式Linux几乎支持所有的嵌入式CPU，同时几乎可以被移植到所有的嵌入式OEM板。可见嵌入式Linux在可预计的未来，将保持着旺盛的生命力。

综上所述，嵌入式Linux具有诸多优点：首先，Linux因其开源的特性，得到了全球的众多专业厂家及编程爱好者的支持；其次，Linux的内核只有约134KB大小，具有执行效率高，更新速度的特点，而且也是可以定制的。第三，Linux系统因其免费的特点，具备价格优势。Linux适用于嵌入式操作系统还体现在，它是一个跨平台的系统，截止目前，有二三十种CPU和硬件平台的支持。具有良好的稳定性，裁剪性，开发和使用的便利性。其移植速度远远超过以前的Java系统，因此越来越多的CPU芯片厂商纷纷加入这一行列。同时，Linux在网络方面的完整性得益于其内核的结构，甚至于提供了以及无线网络，令牌环网(Tokerring)、光纤甚至卫星的支持。嵌入式Linux提供了包括十兆、百兆、千兆的以太网络，完全支持目前应用最为广泛的网络TCP/IP协议。保证了目前使用Linux环境开发的产品，在将来的适用性。

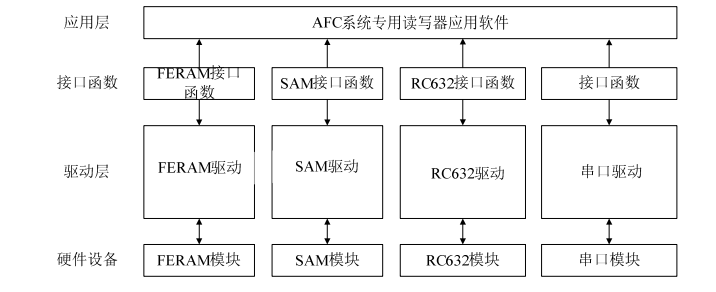
读写器选用Linux系统，并在此系统上开发驱动和应用程序。Linux操作系统有两种空间状态，各个功能硬件设备的驱动位于Linux的内核空间，而其他应用软件在其用户空间。其具体软件架构层次如图所示：



嵌入式 Linux 系统体系结构基于层次化思想进行设计，其分层架构模型如图所示，从底层向上依次为硬件层、OEM、操作系统层和应用层。硬件层是指系 统的硬件组成，包括微处理器、各种外围设备和硬件功能模块；OEM 层位于硬件层和操作系统层之间，为二者的交互提供接口，操作系统可以通过OEM适配层提供的应用接口程序访问具体的硬件，从而避免了直接与硬件进行匹配。操作系统层 位应用程序提供载体，可以根据具体的用户需求定制 Linux 操作系统的组件，保留需要的组件，删除不必要的组件，合理的剪裁可以优化系统的性能。应用层包括 Linux 应用程序和用户应用程序，是用户为特定的系统开发的个性化应用。

**2、 操作系统及底层驱动方案**

根据Linux的层次体系架构，结合AFC系统专用读写器系统的具体情况，可以得出AFC系统专用读写器系统的架构，大多数功能硬件需要开发驱动，除去自身封装有驱动程序的硬件，主要的硬件驱动程序与操作系统以及应用软件的结构关系如图4-2所示，其中不包括LED灯等次要硬件及其驱动程序。



应用软件需要采用C/C++语言开发，代码只采用标准的ANSI函数库和POSIX函数库，这样程序具有很强的可移植性。软件能根据不同的操作系统、开发环境的变化而重新独立编译或移植。

设备驱动程序负责将操作系统的请求传输转化为特定物理设备控制器能够理解的命令，通过读写硬件寄存器实现对硬件设备的控制。设备驱动程序是操作系统和输入输出设备之间的沟通桥梁，它使计算机能够认识和识别这些硬件设备，从而保证硬件设备的正常运行。设备驱动程序是一种内核模块，负责管理硬件设备底层I/O操作，内核通过调用标准接口函数与设备进行交互。

在Linux中，为了简化对设备的管理，所有外围的硬件设备被归结为三类：字符设备、块设备和网络设备。字符设备是指鼠标、键盘、串行口等，存取时没有缓存的设备；块设备是指硬盘设备、FLASH等，读写都有缓存支持，并能随机存取；网络设备在Linux里做专门处理，系统里支持对发送数据和接收数据的缓存，但不能通过简单的读写操作来访问它们，如以太网卡等。除了网络设备外，字符设备和块设备都被映射到Linux文件系统的文件和目录中，访问字符设备和块设备需通过文件系统的调用接口函数open、write、read、close实现。而块设备较字符设备复杂，在块设备上边首先会建立一个磁盘/FLASH文件系统，不同的设备对应的文件系统也不同，如上一章介绍的硬件中NORFLASH采用JFFS2文件系统，NANDFLASH采用YAFFS2文件系统。如图所是Linux设备驱动与整个软硬件系统的关系。



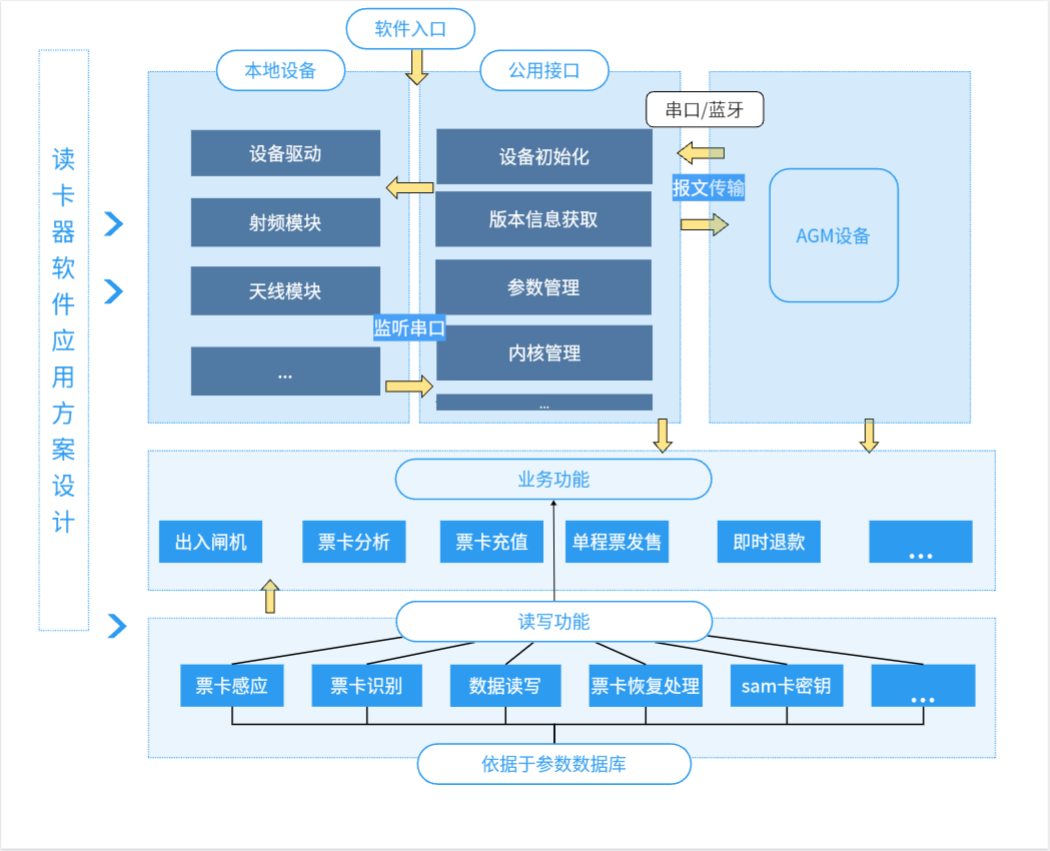
Linux将所有外部设备看成是一类特殊文件，称之为设备文件，设备驱动程序相当于是Linux内核与外部设备之间的接口。设备驱动程序简化了应用程序对硬件的操作，实现应用程序操作硬件时与正常文件操作没有特殊不同。设备驱动程序的主要功能包括：设备初始化和释放；设备的参数设置以及操作设备的接口；实现应用程序与设备文件之间数据的传送和返回请求；实现对设备的监测及纠错处理。

Linux操作系统对所有的设备的操作是通过文件的操作界面进行的，而设备文件的属性由三部分信息组成：文件的类型、主设备号和次设备号。文件类型和主设备号两者共同保证了设备文件的驱动程序唯一性，也就是说主设备号相同的设备使用相同的驱动程序，次设备号只是用于说明目标设备在同类设备的位次。Linux操作系统按所存空间不同分为内核空间和用户空间。内核空间是指在被保护的内存空间上的核心软件，拥有较高的级别，拥有所有访问硬件设备的权限；用户空间则为在用户空间执行其他部分的应用程序。Linux系统实现从用户空间到内核空间的转移主要是通过系统调用和硬件中断完成的，大量的设备驱动程序及相应的应用软件就是采用内核级与用户级的程序一起来完成驱动与访问的。驱动程序是运行于内核空间的，用户空间的应用程序通过文件系统中的一个设备文件与其交互。

**3、读卡器软件应用方案**

通过对AFC系统专用读卡器在实际中应用的研究，发现读卡器软件技术在设计与开发的过程中主要有四个方面。⑴需要将动态库技术应用于系统之中，这种技术能够实现程序空间的有效压缩，从而促进程序执行效率的提高，使其运行具有较高的灵活性。动态库技术的应用能够使其使用更加简便；⑵需要将软件进行参数化处理，参数化主要体现在两个方面，分别为票务规则参数化、软件自身的参数化，通过参数的设置，来进行不同选项的配置，来满足系统在功能方面的需求；⑶需要保证系统数据的安全性，这一点在软件开发全过程中都是需要进行重点考虑的环节；⑷系统具备兼容性，这种性能保证了读卡器软件自身运行具有的时效性，使城市轨道交通是AFC系统在长时间的运行状态下的需求能够得到满足。

轨道交通AFC系统专用读卡器方案要得以实现，必须要从驱动程序与票卡交易两方面的实现入手。驱动程序的主要作用就是转化，当操作系统进行信息请求与传输的过程中，通过转化的作用将请求转化为物理设备控制器，这种特定的控制器能够对请求命令更加方便的进行理解，并对硬件寄存器进行读卡以达到对硬件设备进行控制的目的。票卡交易主要作用是实现各个载体的通信，每一个载体的功能都不同，比如车站进出口的闸机、用于售票的自动与半自动售票机以及自动加值机等等，票卡就是在车站中的这些机器载体中来实现其通讯目的的。

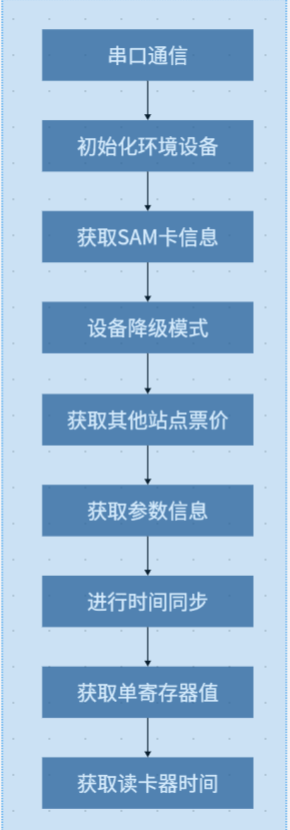


AFC系统专用读写器在城市轨道交通AFC系统中主要应用在闸机(自动检票机)、自动售票机、半自动售票机以及自动加值机上，通过这些载体与票卡进行通讯。AFC系统专用读写器的作用就是监管、控制或者更改票卡里的特定数据从而帮助实现AFC系统对城市轨道交通的票务管理。根据以上设备的功能可知票卡交易的过程有充值、进站和出站等。读写器的功能是与票卡之间实现有效可靠的交易，所有读写器的应用层软件的设计就是围绕读写器这一功能进行的。

票卡交易的最主要组成部分是检票过程，检票时AFC系统专用读写器读取乘客所持票卡的信息，根据卡内文件数据判断卡是否在系统内有效。在进站时，AFC系统专用读写器向交易文件中写入入站站信息和入站时间信息，单程票和储值票均可被AFC系统专用读写器处理。在出站时读写器读票卡交易文件中的入站时间信息，查看是否超程或超时；读取入站站信息计算乘车区间和票价，读智能卡的钱包值文件，并从中扣除相应的金额(或次数)。以储值票为例，分析票卡交易过程。首先分析轨道交通AFC系统储值票应用的实际情况，分析储值票票卡数据结构；其次，结合票卡数据结构，研究交易过程的防撕裂操作；最后，考虑可能出现的各种情况，分析票卡交易的具体流程。

### 软件方案模块设计

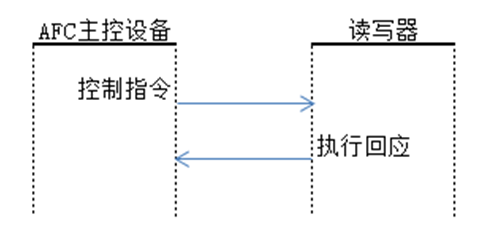
**1、公用接口子模块方案介绍**



（1）串口通信

UART串口简介UART全称是通用异步收发传输器（串口顾名思义是数据串行接口，即数据的传输是一位接一位传输，属于一种串行的数据总线，属于异步通讯，同时支持全双工数据传输（全双工数据传输：允许发送数据和接收数据在同一时刻发生）。除了UART，另外还有一种叫USART，全称是通用同步/异步串行接收/发送器，USART比UART多了同步通信功能，但是百分之90的工程应用中不会应用该同步功能，都是将USART当做UART使用，即采取异步串行通讯。一般开发板或者产品中都会将UART串口标为serial或COM

所有业务逻辑由非接触读卡器实现。当需要对读写器进行设置或读写卡片时，终端向读写器发送相应指令，由读写器返回操作结果。若终端未发起支付交易命令，读卡器应暂时禁止寻卡功能。除了从读卡器接收卡片信息外，终端还可能需要执行其他功能，因此必须确保报文流程不会显著的增加交易时间。

****

（2）设备初始化

初始化读卡器的各硬件模块。根据接口传入的参数，加载设备所需参数，申请内部资源设备要调用读写器处理票卡前必须先执行此调用。设备在初始化之前应先调用‘Config\_Parameter’配置好设备必须的参数.

（3）版本信息获取

读取读写器对应的版本信息。

（4）获取SAM卡信息

读取读写器对应类型的SAM卡相关信息。

（5）设备降级模式

降级模式是统称,它是指在本级行车控制模式下不能工作时,自动或手动转换到下一级行车控制模式下,并维持运营的方式,一般来说,降级模式是比上一级模式功能降低,级别降低的一种方式。有ACT系统降级,ATO/ATP系统降级以及最不希望出现的电话闭塞法。因为现在的地铁网络十分发达,同时也就意味着十分“脆弱”,任何一个设备的故障都有可能影响整个系统

当SLE（本地无线交换机）接收到SC下发的降级模式通知或降级模式控制命令后，SLE将通知或控制命令的内容填入结构pDegradeCmd后调用本接口，本接口被调用后，API不对pDegradeCmd信息做永久记录，API从内存释放后pDegradeCmd信息丢失，API被再次载入内存时，上次载入时的pDegradeCmd信息被忽略，除非上位机软件再次调用本接口。

上位机软件应永久记录pDegradeCmd相关信息直至该信息失效（当前时间超出该降级消息的敏感期），并在每次调用完初始化接口后调用本接口。

当有新的降级模式参数后，API将以降级模式参数为准，忽略之前所有调用本接口所产生的降级。

（6）获取其他站点到本站的票价

获取当前时间其它站到本站的票价，做付费区无进站码更新时可能需要额外补款，可以通过本接口配合。

（8）获取参数信息

读取读写器相应参数信息。

（9）时间同步

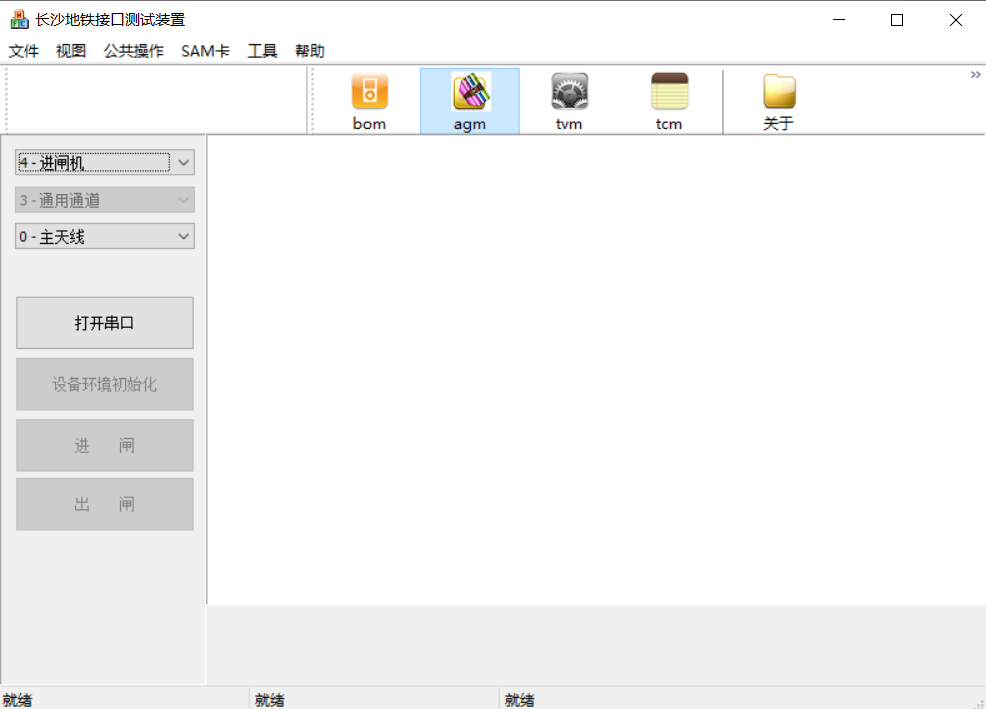
同步读写器时钟。

（10）获取单个寄存器值

读取当前设备固定地址的寄存器

（11）获取读卡器当前时间

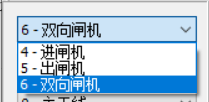
**2、 AGM机业务功能模块方案介绍**



（1）通道类型设置模块

设定读写器当前所属通道类型，此函数只适用于闸机，闸机车票处理流程中读写器根据此通道类型去处理对应类型的车票。

上位机应用软件调用完读写器初始化接口后紧接着调用此接口函数，如不调用此接口函数，在入/出闸处理中做为正常通道处理。



（2）入闸处理模块

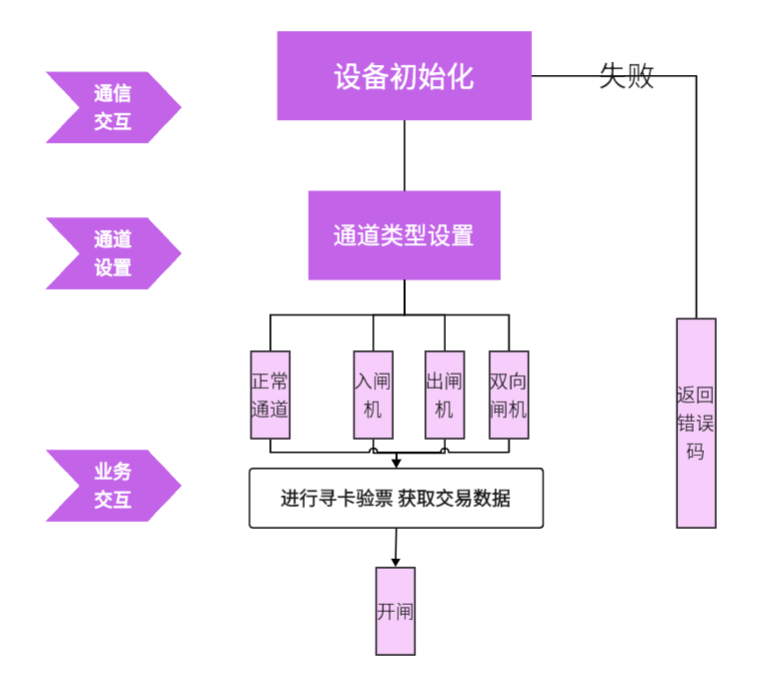
在入口闸机中调用此接口，成功进行交易后返回进闸交易数据，否则返回操作错误状态码。如果扩充保留字段为0x11,0x00，代表为电子车票业务，电子票入闸接口仅返回ITP验证电子票合法性的票卡结构，交易数据由开闸处理接口产生。

（4）出闸处理模块

在出口闸机中调用此接口，根据应用情况指定主天线处理储值票或是副天线处理单程票，成功进行交易后返回出闸交易数据，否则返回操作错误状态码。如果扩充保留字段为0x11,0x00，代表为电子车票业务，电子票出闸接口仅返回ITP验证电子票合法性的票卡结构，交易数据由开闸处理接口产生。

（5）开闸处理模块

当调用完入闸和出闸接口后，如果返回为电子票交易时，需要在调用此接口，获得开门指令以及交易数据。



**3、票卡读写模块方案介绍**

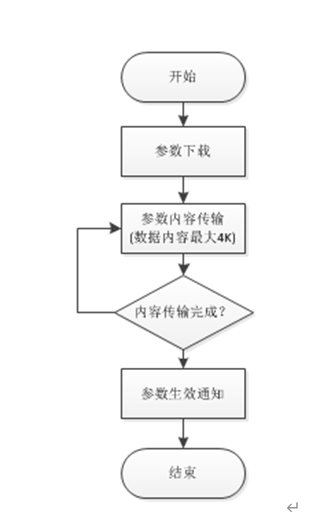
票卡识别包括寻卡、解析码头、得到密钥、先解密二维码转成明文、校验票卡合法性、明文分析。

**4、运营参数模块方案介绍**

1、进行参数下载

当参数表有更新时，ECU上层应用程序将参数表文件下载到本地存储器中，并调用此接口函数通知API启用此参数文件。

参数下载流程:



参数下载生效通知

参数下载开始前和参数下载完成后调用。

参数内容传输

将参数文件分包下发到读写器，每次下发最大数据内容为4Kbyte。

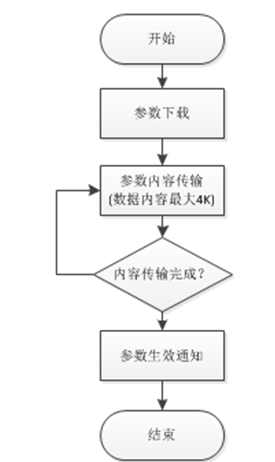
2、获取参数信息

获取当前启用指定参数表的文件名，不包含全路径。

3、ITP参数下载

当ITP参数表有更新时，ECU上层应用程序将ITP参数表文件下载到本地存储器中，并调用此接口函数通知API启用此参数文件。

参数下载流程：



ITP参数下载生效通知

ITP参数下载开始前和参数下载完成后调用。

ITP参数内容传输

将ITP参数文件分包下发到读写器，每次下发最大数据内容为4Kbyte。

4、获取ITP参数信息

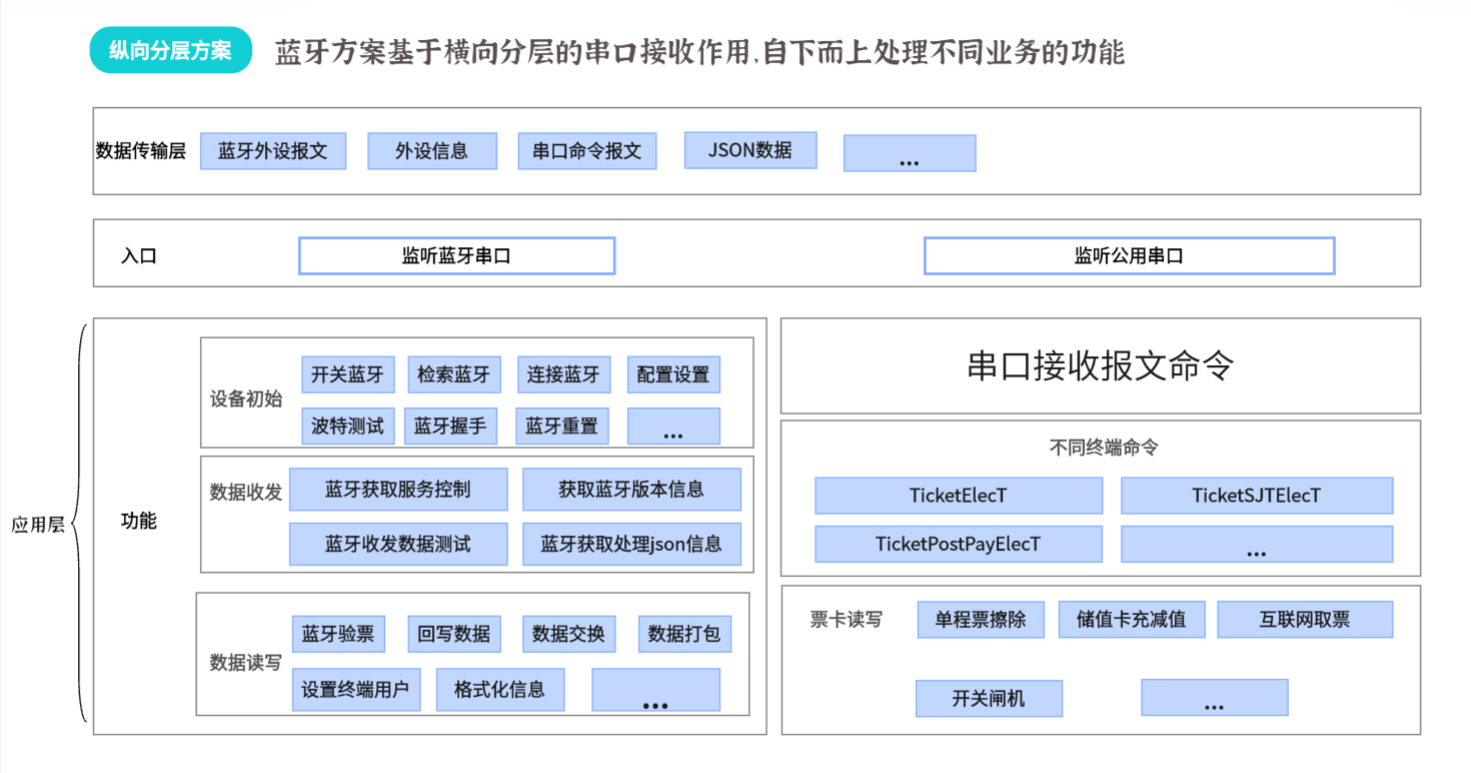
获取当前启用指定ITP参数表的文件名，不包含全路径。

**5、 蓝牙通信模块方案介绍**

蓝牙是一种支持设备之间距离通信的无线电技术，应用在移动电话、ＰＤＡ、车载蓝牙、无线耳机、笔记本电脑等设备中，使用便捷。分散式网络结构、快跳频、短包技术等是蓝牙主要技术特点，蓝牙支持点对点、点对多点通信，可采用时分全双工进行数据传输。蓝牙低能耗（BluetoothLowEnergy，或称BluetoothLE、BLE，旧商标BluetoothSmart）也称低功耗蓝牙，是[蓝牙技术联盟](https://baike.baidu.com/item/%E8%93%9D%E7%89%99%E6%8A%80%E6%9C%AF%E8%81%94%E7%9B%9F/9308756?fromModule=lemma_inlink" \t "_blank)设计和销售的一种[个人局域网](https://baike.baidu.com/item/%E4%B8%AA%E4%BA%BA%E5%B1%80%E5%9F%9F%E7%BD%91?fromModule=lemma_inlink)技术，旨在用于[医疗保健](https://baike.baidu.com/item/%E5%8C%BB%E7%96%97%E4%BF%9D%E5%81%A5/5996450?fromModule=lemma_inlink)、运动健身、[信标](https://baike.baidu.com/item/%E4%BF%A1%E6%A0%87/13781016?fromModule=lemma_inlink)、[安防](https://baike.baidu.com/item/%E5%AE%89%E9%98%B2/412165?fromModule=lemma_inlink)、家庭[娱乐](https://baike.baidu.com/item/%E5%A8%B1%E4%B9%90/279534?fromModule=lemma_inlink)等领域的新兴应用。相较经典蓝牙，低功耗蓝牙旨在保持同等通信范围的同时显著降低功耗和成本。

由于蓝牙模块的通信范围较大，速度较快，广泛应用于智能手机、笔记本电脑、无线耳机、汽车等设备。基于读卡器文档对智能设备交互的需求，现提出蓝牙通信模块方案。该蓝牙方案能满足读卡器对周围智能设备高效、低时延、多用户的通信情景、能满足多种不同设备的接入和数据共享、能满足行业内通信标准协议扽等。欲通过读卡器蓝牙模块实现与外设的通信，蓝牙模块要提供了一系列需要的API，如打开蓝牙适配器、搜索蓝牙、连接蓝牙等等。

蓝牙方案设计如下图所示：



读卡器的蓝牙通信模块方案主要由两个串口监听组成。首先是蓝牙硬件模块接口。可对蓝牙进行设备初始化、数据接收、数据读写等。可按照协议规约进行识别、授权、配对和管理，以保证蓝牙通信的安全性和可靠性。其次是公用串口用于实现不同的业务功能。则利用主从链路通信，实现无线远距离稳定传输，确保快速采集配上位机终端的关键数据。以满足票卡读写、互联网取票、闸机控制等功能。

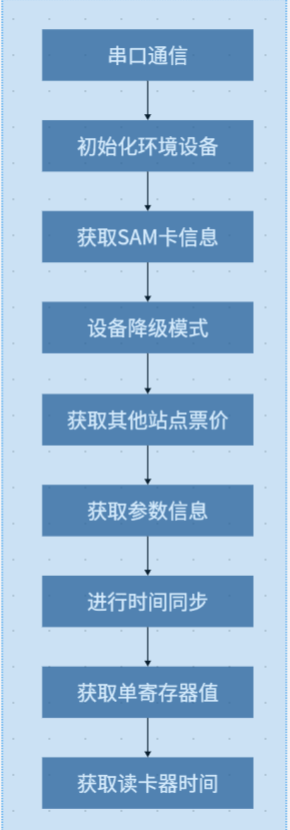
### 方案设计标准

### 方案总结

# 概要设计

## 读写器软件概要设计

3.1.1 公用接口子模块设计

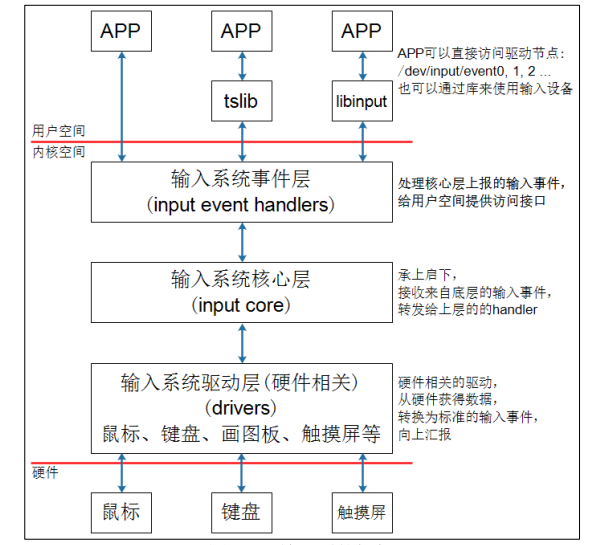


**（1）串口通信**

UART 串口最精简的连接是TTL电平三线连接UARTx\_TXD：用于发送数据，应连接到接收设备的UARTx\_RXD引脚上；UARTx\_RXD：用于接收数据，应连接到发送设备的 UARTx\_TXD 引脚上； GND：为双方提供一个相同的参考电平。

常见的输入设备有键盘、鼠标、遥控杆、书写板、触摸屏等等,用户通过这些输入设备与 Linux 系统进行数据交换。

Linux 系统为了统一管理这些输入设备，实现了一套能兼容所有输入设备的框架：输入系统。驱动开发人员基于这套框架开发出程序，应用开发人员就可以使用统一的 API 去使用设备。



设用户程序直接访问/dev/xxx设备节点，或者使用tslib访问设备节点，数据的流程如下：

（1）APP发起读操作，若无数据则休眠；

（2）用户操作设备，硬件上产生中断；

（3）输入系统驱动层对应的驱动程序处理中断：读取到数据，转换为标准的输入事件，向核心层汇报。所谓输入事件就是一个“structinput\_event”结构体。

（5）核心层可以决定把输入事件转发给上面哪个handler来处理：从handler的名字来看，它就是用来处输入操作的。有多种handler，比如：evdev\_handler、kbd\_handler、joydev\_handler等等。最常用的是evdev\_handler：它只是把input\_event结构体保存在内核buffer等，APP来读取时就原原本本地返回。它支持多个APP同时访问输入设备，每个APP都可以获得同一份输入事件。当APP正在等待数据时，evdev\_handler会把它唤醒，这样APP就可以返回数据。

（6）APP对输入事件的处理：APP获得数据的方法有2种：直接访问设备节点(比如/dev/xxx)，或者通过tslib、libinput这类库来间接访问设备节点。这些库简化了对数据的处理。

查询方式：APP调用open函数时，传入“O\_NONBLOCK”表示“非阻塞”。APP调用read函数读取数据时，如果驱动程序中有数据，那么APP的read函数会返回数据，否则也会立刻返回错误。

休眠-唤醒方式APP调用open函数时，不要传入“O\_NONBLOCK”。APP调用read函数读取数据时，如果驱动程序中有数据，那么APP的read函数会返回数据；否则APP就会在内核态休眠，当有数据时驱动程序会把APP唤醒，read函数恢复执行并返回数据给APP。

POLL/SELECT方式：功能介绍POLL机制、SELECT机制是完全一样的，只是APP接口函数不一样。简单地说，它们就是“定个闹钟”：在调用poll、select函数时可以传入“超时时间”。在这段时间内，条件合适时(比如有数据可读、有空间可写)就会立刻返回，否则等到“超时时间”结束时返回错误。用法如下。APP先调用open函数时。APP不是直接调用read函数，而是先调用poll或select函数，这2个函数中可以传入“超时时间”。它们的作用是：如果驱动程序中有数据，则立刻返回；否则就休眠。在休眠期间，如果有人操作了硬件，驱动程序获得数据后就会把APP唤醒，导致poll或select立刻返回；如果在“超时时间”内无人操作硬件，则时间到后poll或select函数也会返回。APP可以根据函数的返回值判断返回原因：有数据？无数据超时返回？APP根据poll或select的返回值判断有数据之后，就调用read函数读取数据时，这时就会立刻获得数据。poll/select函数可以监测多个文件，可以监测多种事件：表7-1poll/select监测事件

异步通知方式：所谓同步，就是“你慢我等你”。那么异步就是：你慢那你就自己玩，我做自己的事去了，有情况再通知我。所谓异步通知，就是APP可以忙自己的事，当驱动程序用数据时它会主动给APP发信号，这会导致APP执行信号处理函数。Linux系统中也有很多信号，在Linux内核源文件include\uapi\asmgeneric\signal.h中，有很多信号的宏定义：驱动程序通知APP时，它会发出“SIGIO”这个信号，表示有“IO事件”要处理。

应用编程：应用程序要做的事情有这几件：

①编写信号处理函数：

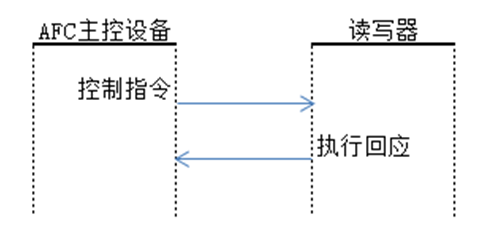
②注册信号处理函数

③打开驱动：fd=open(argv[1],O\_RDWR);

④把进程ID告诉驱动：

⑤使能驱动的FASYNC功能：

所有业务逻辑由非接触读卡器实现。当需要对读写器进行设置或读写卡片时，终端向读写器发送相应指令，由读写器返回操作结果。若终端未发起支付交易命令，读卡器应暂时禁止寻卡功能。除了从读卡器接收卡片信息外，终端还可能需要执行其他功能，因此必须确保报文流程不会显著的增加交易时间。

****

**（2）设备初始化**

初始化读卡器的各硬件模块。根据接口传入的参数，加载设备所需参数，申请内部资源设备要调用读写器处理票卡前必须先执行此调用。设备在初始化之前应先调用‘Config\_Parameter’配置好设备必须的参数.

**（3）版本信息获取**

读取读写器对应的版本信息。

**（4）获取SAM卡信息**

读取读写器对应类型的SAM卡相关信息。

**（5）设备降级模式**

降级模式是统称,它是指在本级行车控制模式下不能工作时,自动或手动转换到下一级行车控制模式下,并维持运营的方式,一般来说,降级模式是比上一级模式功能降低,级别降低的一种方式。有ACT系统降级,ATO/ATP系统降级以及最不希望出现的电话闭塞法。因为现在的地铁网络十分发达,同时也就意味着十分“脆弱”,任何一个设备的故障都有可能影响整个系统

当SLE（本地无线交换机）接收到SC下发的降级模式通知或降级模式控制命令后，SLE将通知或控制命令的内容填入结构pDegradeCmd后调用本接口，本接口被调用后，API不对pDegradeCmd信息做永久记录，API从内存释放后pDegradeCmd信息丢失，API被再次载入内存时，上次载入时的pDegradeCmd信息被忽略，除非上位机软件再次调用本接口。

上位机软件应永久记录pDegradeCmd相关信息直至该信息失效（当前时间超出该降级消息的敏感期），并在每次调用完初始化接口后调用本接口。

当有新的降级模式参数后，API将以降级模式参数为准，忽略之前所有调用本接口所产生的降级。

**（6）获取其他站点到本站的票价**

获取当前时间其它站到本站的票价，做付费区无进站码更新时可能需要额外补款，可以通过本接口配合。

**（7）获取参数信息**

读取读写器相应参数信息。

**（8）时间同步**

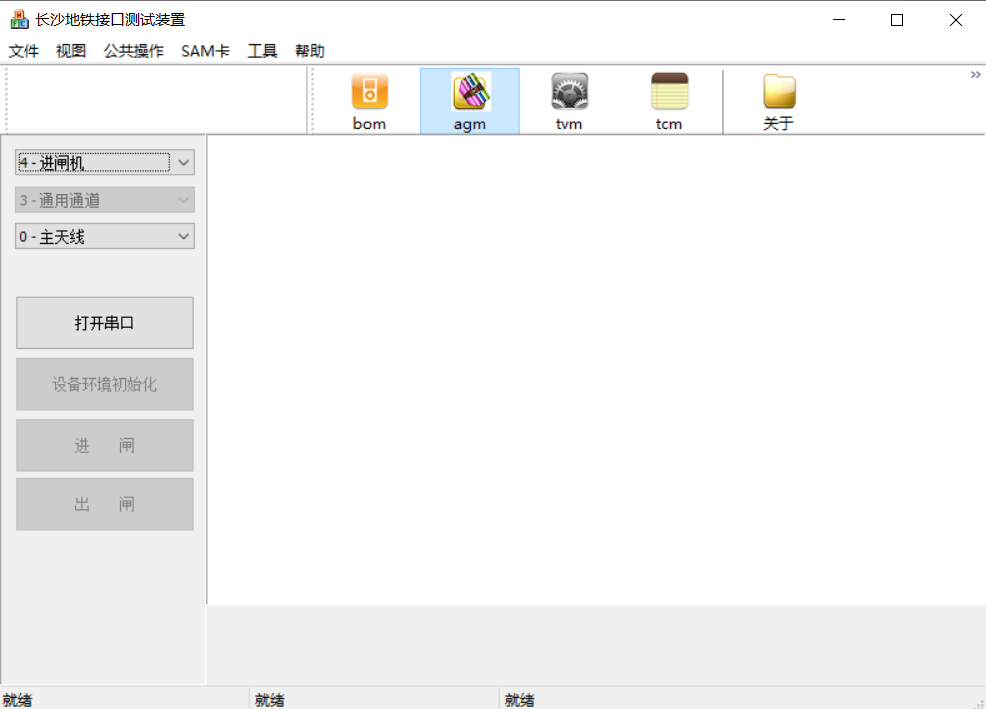
同步读写器时钟。

**（9）获取单个寄存器值**

读取当前设备固定地址的寄存器

**（10）获取读卡器当前时间**

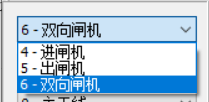
3.1.2 AGM机业务功能模块介绍



**1、通道类型设置模块**

设定读写器当前所属通道类型，此函数只适用于闸机，闸机车票处理流程中读写器根据此通道类型去处理对应类型的车票。

上位机应用软件调用完读写器初始化接口后紧接着调用此接口函数，如不调用此接口函数，在入/出闸处理中做为正常通道处理。



**2、入闸处理模块**

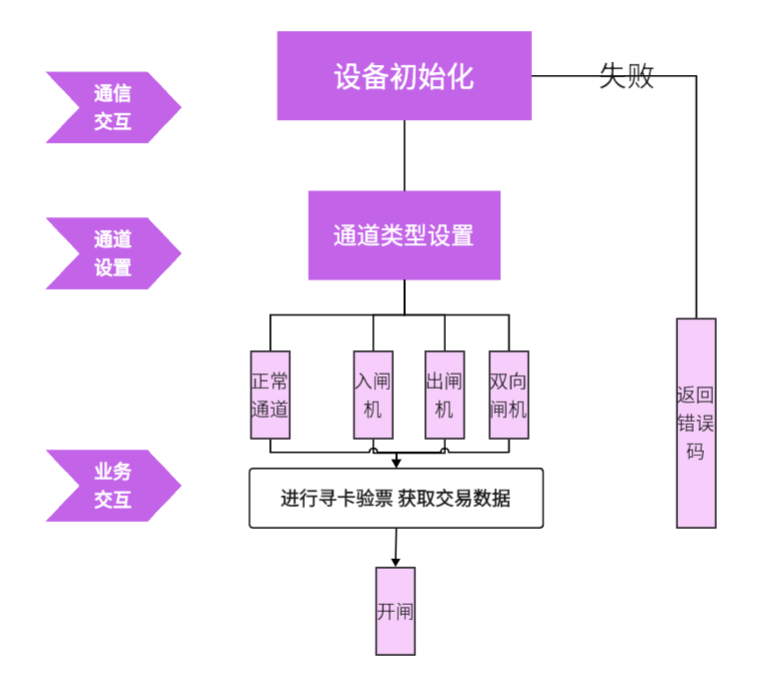
在入口闸机中调用此接口，成功进行交易后返回进闸交易数据，否则返回操作错误状态码。如果扩充保留字段为0x11,0x00，代表为电子车票业务，电子票入闸接口仅返回ITP验证电子票合法性的票卡结构，交易数据由开闸处理接口产生。

**3、出闸处理模块**

在出口闸机中调用此接口，根据应用情况指定主天线处理储值票或是副天线处理单程票，成功进行交易后返回出闸交易数据，否则返回操作错误状态码。如果扩充保留字段为0x11,0x00，代表为电子车票业务，电子票出闸接口仅返回ITP验证电子票合法性的票卡结构，交易数据由开闸处理接口产生。

**4、开闸处理模块**

当调用完入闸和出闸接口后，如果返回为电子票交易时，需要在调用此接口，获得开门指令以及交易数据。



3.1.3 票卡读写模块介绍

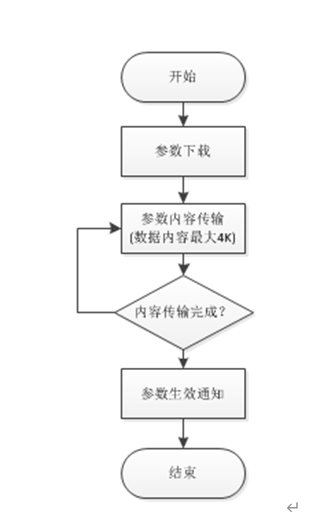
票卡识别包括寻卡、解析码头、得到密钥、先解密二维码转成明文、校验票卡合法性、明文分析。

3.1.4运营参数模块介绍

1、进行参数下载

当参数表有更新时，ECU上层应用程序将参数表文件下载到本地存储器中，并调用此接口函数通知API启用此参数文件。

参数下载流程:



参数下载生效通知

参数下载开始前和参数下载完成后调用。

参数内容传输

将参数文件分包下发到读写器，每次下发最大数据内容为4Kbyte。

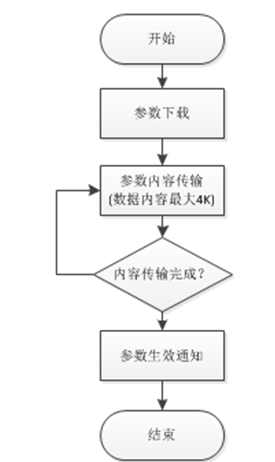
2、获取参数信息

获取当前启用指定参数表的文件名，不包含全路径。

3、ITP参数下载

当ITP参数表有更新时，ECU上层应用程序将ITP参数表文件下载到本地存储器中，并调用此接口函数通知API启用此参数文件。

参数下载流程：



ITP参数下载生效通知

ITP参数下载开始前和参数下载完成后调用。

ITP参数内容传输

将ITP参数文件分包下发到读写器，每次下发最大数据内容为4Kbyte。

4、获取ITP参数信息

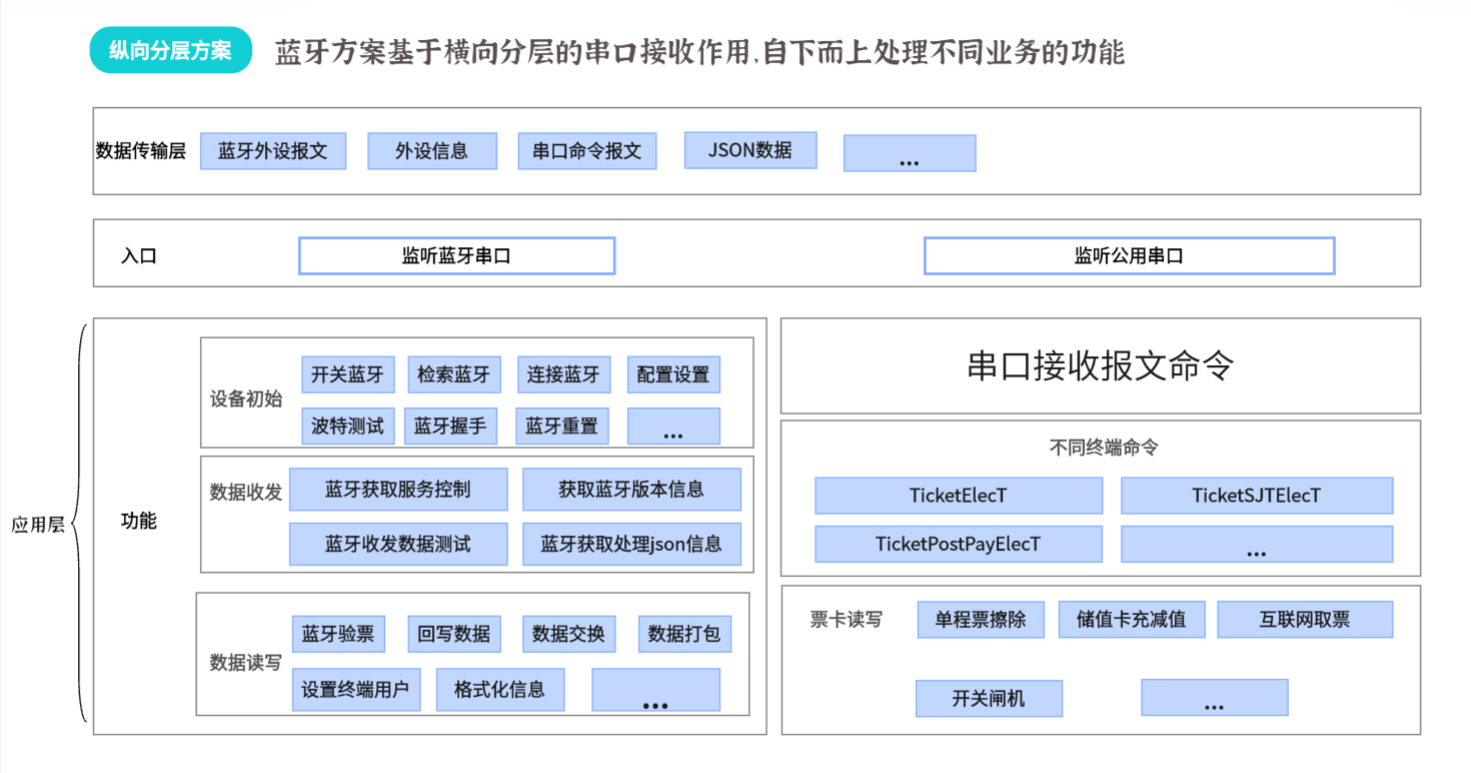
获取当前启用指定ITP参数表的文件名，不包含全路径。

3.1.5蓝牙通信模块介绍

蓝牙是一种支持设备之间距离通信的无线电技术，应用在移动电话、ＰＤＡ、车载蓝牙、无线耳机、笔记本电脑等设备中，使用便捷。分散式网络结构、快跳频、短包技术等是蓝牙主要技术特点，蓝牙支持点对点、点对多点通信，可采用时分全双工进行数据传输。蓝牙低能耗（BluetoothLowEnergy，或称BluetoothLE、BLE，旧商标BluetoothSmart）也称低功耗蓝牙，是[蓝牙技术联盟](https://baike.baidu.com/item/%E8%93%9D%E7%89%99%E6%8A%80%E6%9C%AF%E8%81%94%E7%9B%9F/9308756?fromModule=lemma_inlink" \t "_blank)设计和销售的一种[个人局域网](https://baike.baidu.com/item/%E4%B8%AA%E4%BA%BA%E5%B1%80%E5%9F%9F%E7%BD%91?fromModule=lemma_inlink)技术，旨在用于[医疗保健](https://baike.baidu.com/item/%E5%8C%BB%E7%96%97%E4%BF%9D%E5%81%A5/5996450?fromModule=lemma_inlink)、运动健身、[信标](https://baike.baidu.com/item/%E4%BF%A1%E6%A0%87/13781016?fromModule=lemma_inlink)、[安防](https://baike.baidu.com/item/%E5%AE%89%E9%98%B2/412165?fromModule=lemma_inlink)、家庭[娱乐](https://baike.baidu.com/item/%E5%A8%B1%E4%B9%90/279534?fromModule=lemma_inlink)等领域的新兴应用。相较经典蓝牙，低功耗蓝牙旨在保持同等通信范围的同时显著降低功耗和成本。

由于蓝牙模块的通信范围较大，速度较快，广泛应用于智能手机、笔记本电脑、无线耳机、汽车等设备。基于读卡器文档对智能设备交互的需求，现提出蓝牙通信模块方案。该蓝牙方案能满足读卡器对周围智能设备高效、低时延、多用户的通信情景、能满足多种不同设备的接入和数据共享、能满足行业内通信标准协议扽等。欲通过读卡器蓝牙模块实现与外设的通信，蓝牙模块要提供了一系列需要的API，如打开蓝牙适配器、搜索蓝牙、连接蓝牙等等。

蓝牙方案设计如下图所示：



读卡器的蓝牙通信模块方案主要由两个串口监听组成。首先是蓝牙硬件模块接口。可对蓝牙进行设备初始化、数据接收、数据读写等。可按照协议规约进行识别、授权、配对和管理，以保证蓝牙通信的安全性和可靠性。其次是公用串口用于实现不同的业务功能。则利用主从链路通信，实现无线远距离稳定传输，确保快速采集配上位机终端的关键数据。以满足票卡读写、互联网取票、闸机控制等功能。

## 数据库设计

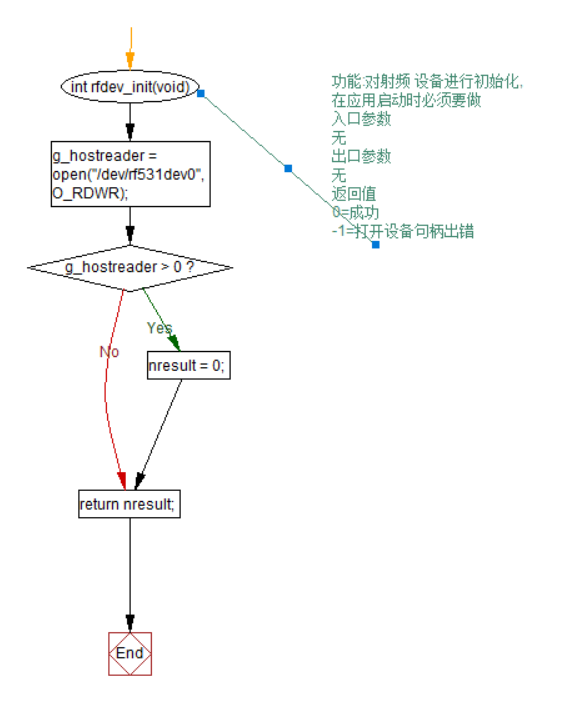
### 应用数据调用图

### 数据模型设计

# 详细设计

**4.1 软件初始模块详细设计**

1、射频设备初始化



2、Sam卡设备初始化

**4.2公用接口模块详细设计**

4.2.1串口通信编程

串口初始化函数

int libserial\_init\_local(char \*dev, int speed, int parity, int databits, int stopbits, int hwf, int swf)

串口发送数据报文函数

int libserial\_send\_package\_local(uchar \*buf, const int buflen);

串口接收数据报文函数

int libserial\_recv\_package\_local(const int maxrecv, int timeout, uchar \*buf);

1、串口设置函数

在Linux系统中，操作设备统一接口就是：open/ioctl/read/write。对于UART，又在ioctl之上封装了很多函数，主要是用来设置行规程。所以对于UART，编程的套路就是：open；设置行规程，比如波特率、数据位、停止位、检验位、RAW模式、一有数据就返回；read/write；怎么设置行规程？行规程的参数用结构体termios来表示，可以参考Linux串口—structtermios结构体：

//=================================================================================================

// 串口初始化函数。

// dev：串口设备名，串口1是"/dev/ttyS0"，串口2是"/dev/ttyS1"，依此类推。

// speed：串口波特率，可以是230400, 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1200, 300, 0。

// parity：奇偶校验。值为'N','E','O','S'。

// databits：数据位，值是5、6、7或者8。

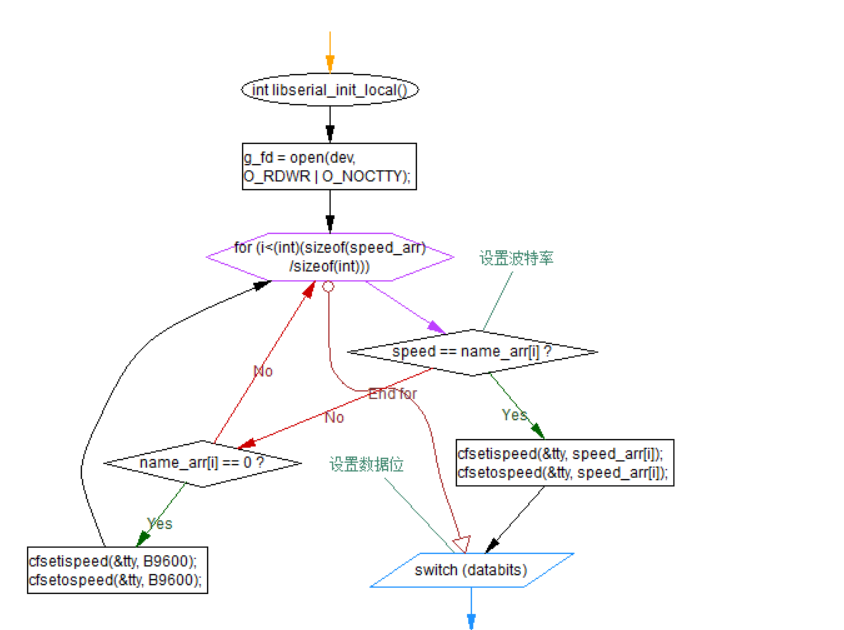
// stopbits：停止位，值是1或者2。

// hwf：硬件流控制。1为打开，0为关闭。

// swf：软件流控制。1为打开，0为关闭。

//==================================================================================================

int libserial\_init\_local(char \*dev, int speed, int parity, int databits, int stopbits, int hwf, int swf)



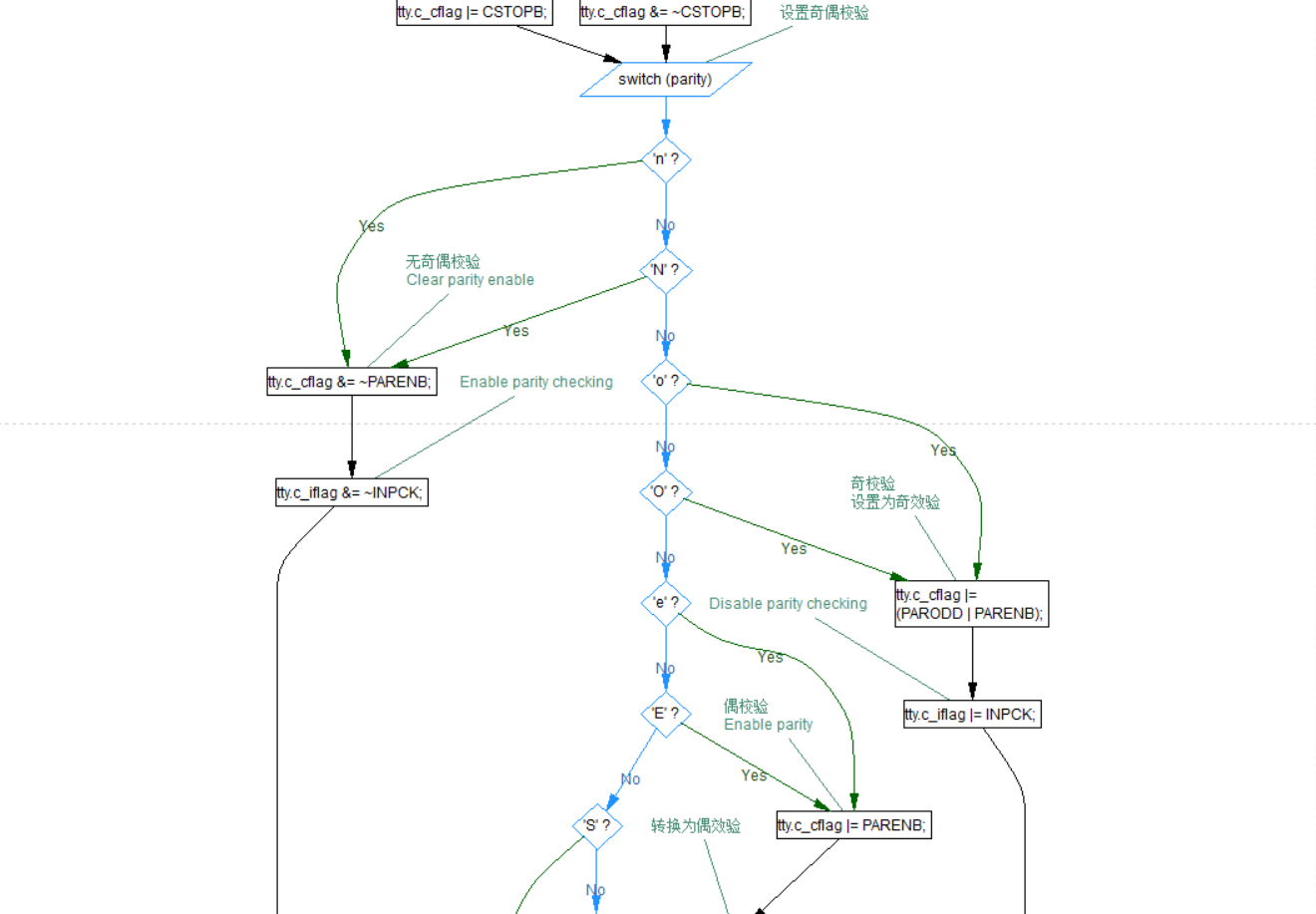
2、UART通讯数据格式：UART之间为何能够准确可靠的的发送和接收数据？首先我们需要约定好传输速率（每一秒传输的数据位数，即波特率），一般选择9600,19200,115200等。确定好传输速率后，我们还需要确定传输数据的格式，UART串口通信的数据包以帧为单位，常用的帧结构为：1位起始位+8位数据位+1位奇偶校验位（可选）+1位停止位。

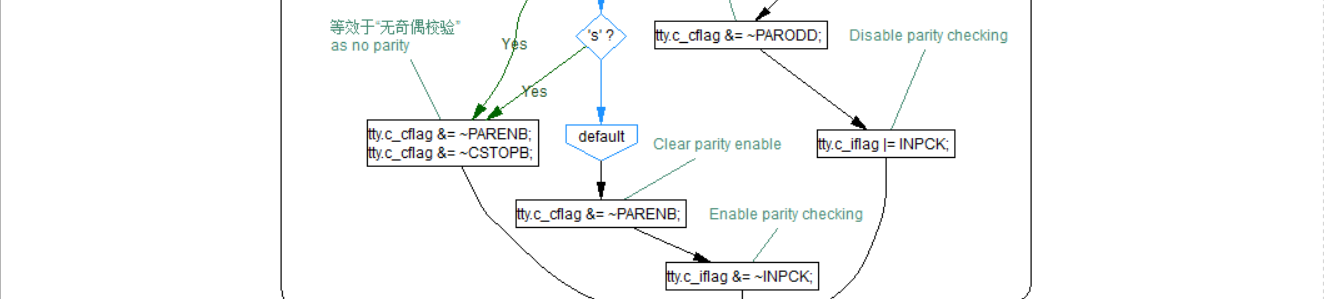
1)空闲位:平时没有数据时，数据线为高电平（逻辑1）；

2)起始位:当需要发送数据时，UART将改变UARTx\_TXD的状态，变为低电平，即为上图中的起始位（逻辑0）；

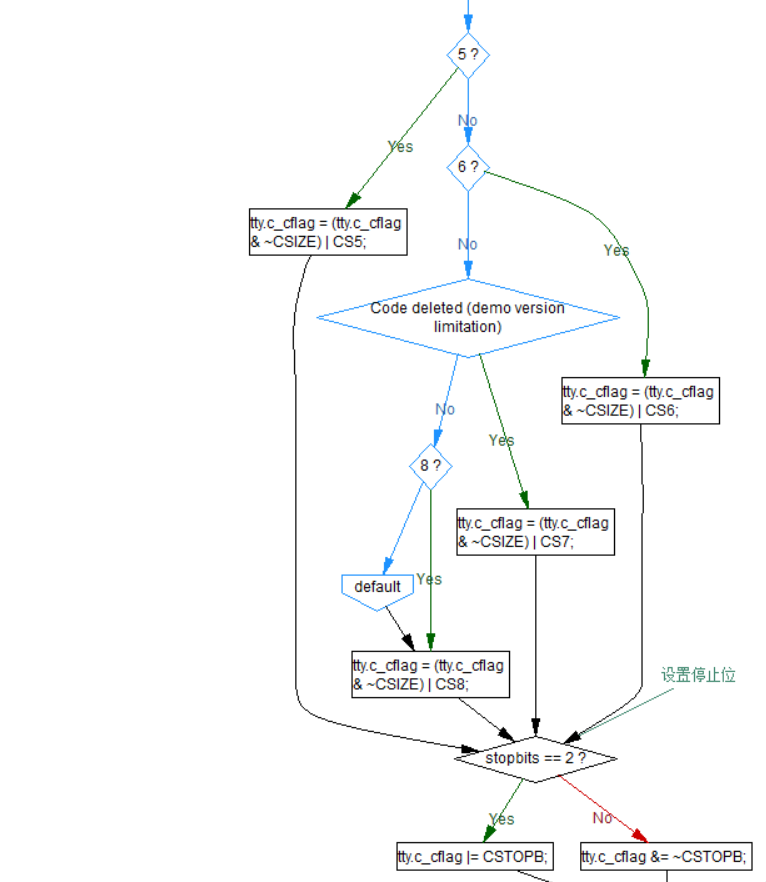
3)数据位:可以有5、6、7或8位的数据，一般我们是按字节（8位）传输数据，发送方一位一位的改变数据线上的状态（高电平或低电平）将它们发送出去，传输数据时先传最低位，最后传送最高位。字符’A’的8位二进制字符是01000001，先发送最低位bit0，其值为1；再发送bit1，其值为0，如此继续；最后发送最高位bit7，其值为0。

4)奇偶校验位如果使用了奇偶校验功能，有效数据位发送完毕后，还要发送1个校验位（奇偶校验位）。有两种校验方法：奇校验，偶校验-------数据位连同校验位中，“1”的数目等于奇数或偶数。奇偶校验只能检错，不能纠错的。而且只能检测1位误码，检测出有错后只能要求重发，没法纠正的。上图中使用的是偶较验，即8个数据位和1个校验位中，一共有偶数个“1”：2个。

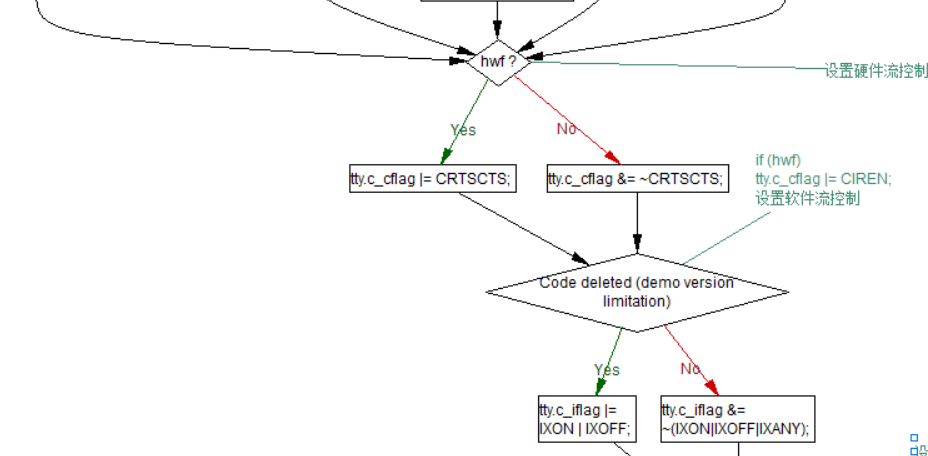


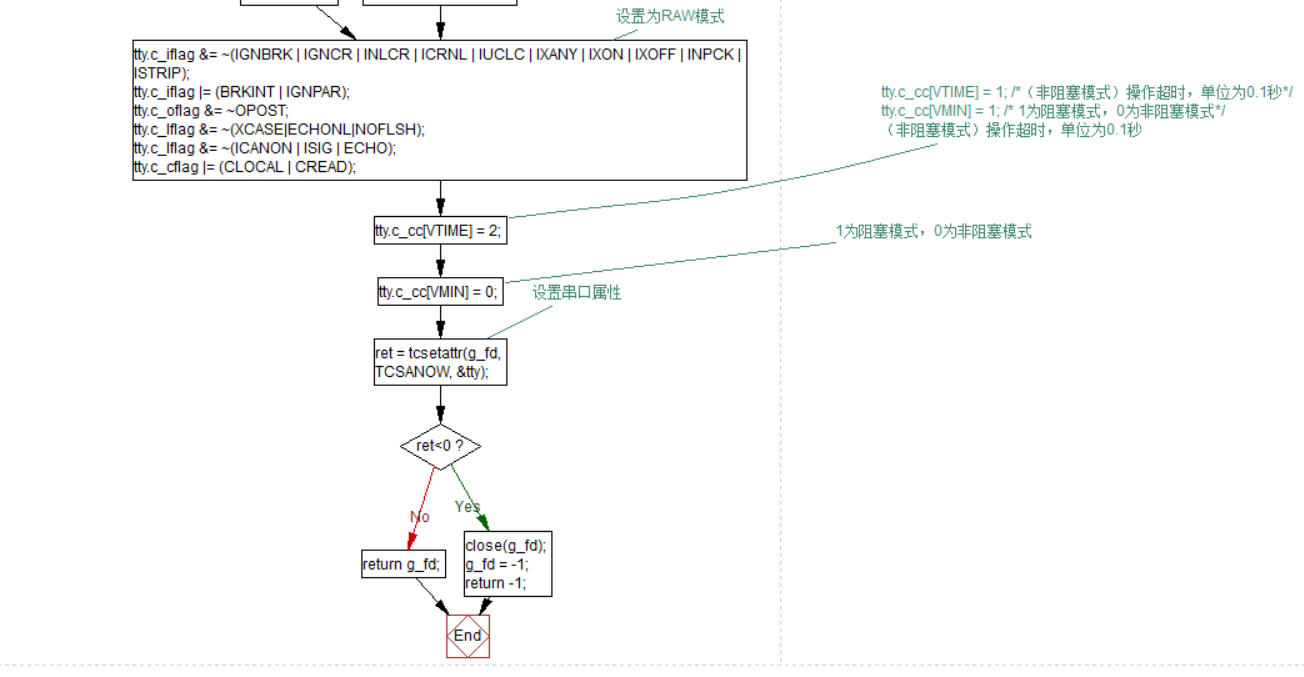


5)停止位停止位（逻辑1）用来表示当前数据传输完毕。停止位的长度有三种：1位，1.5位，2位，通常我们选择1位即可。



6）设置软流控制





2、串口数据发函数

发数据格式如下表所列：



//========================================================

//功能:发送数据报文

//入口参数

// buf=发送缓冲

// buflen=数据长度

//出口参数

// 无

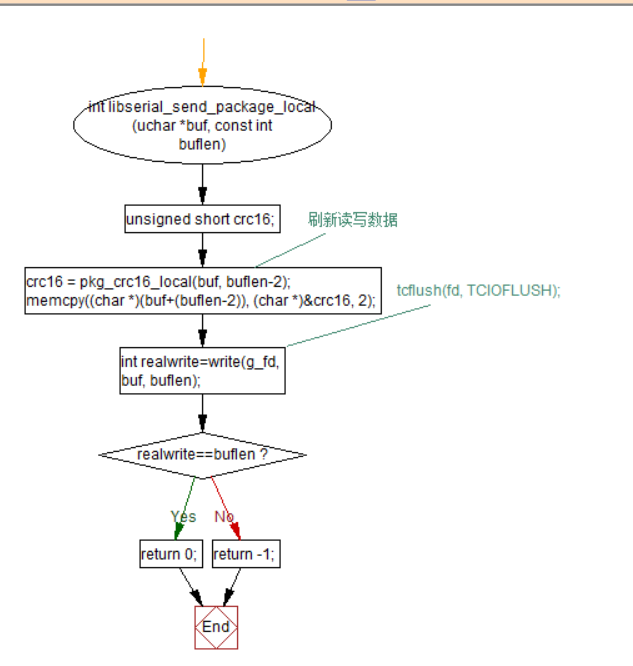
//返回值

// 0=发送完成

// -1=发送失败

//========================================================

int libserial\_send\_package\_local(uchar \*buf, const int buflen);



3、串口收数据

如下表所列：



//功能:接收数据报文

//入口参数

// maxrecv=接收缓冲大小

// timeout=超时时间，单位为100ms

//出口参数

// buf=接收缓冲

//返回值

// 0=执行无返回

// -1=接收超时

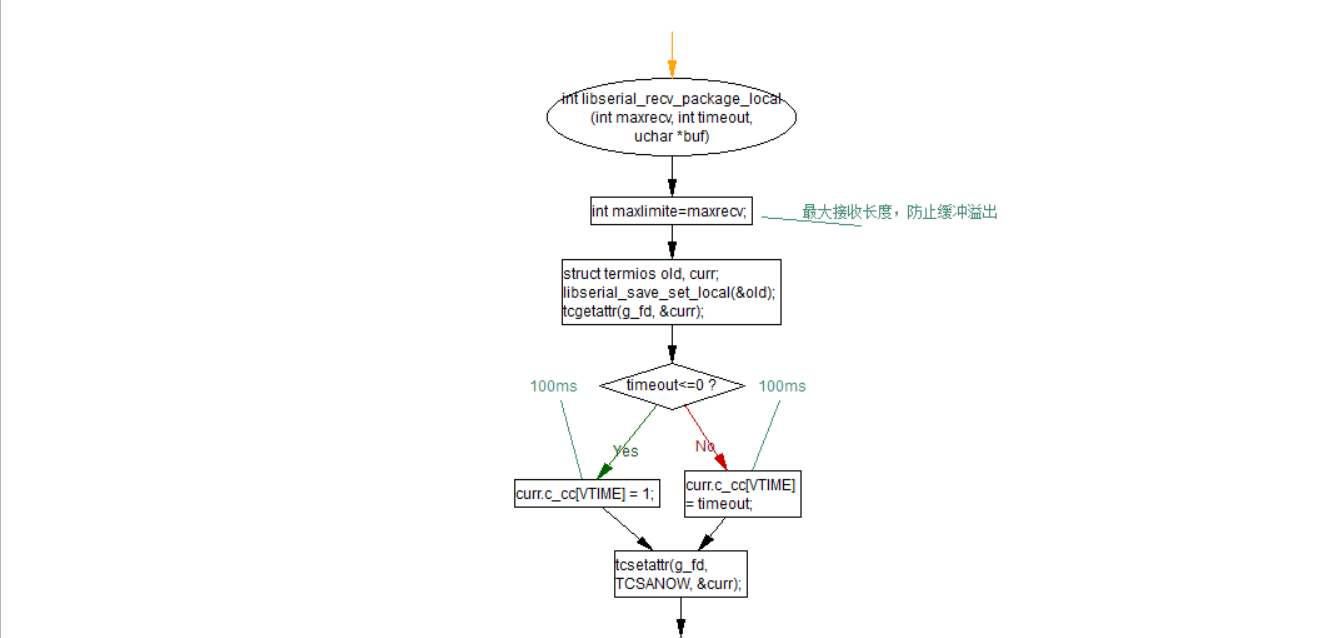
// -2=接收缓冲区满

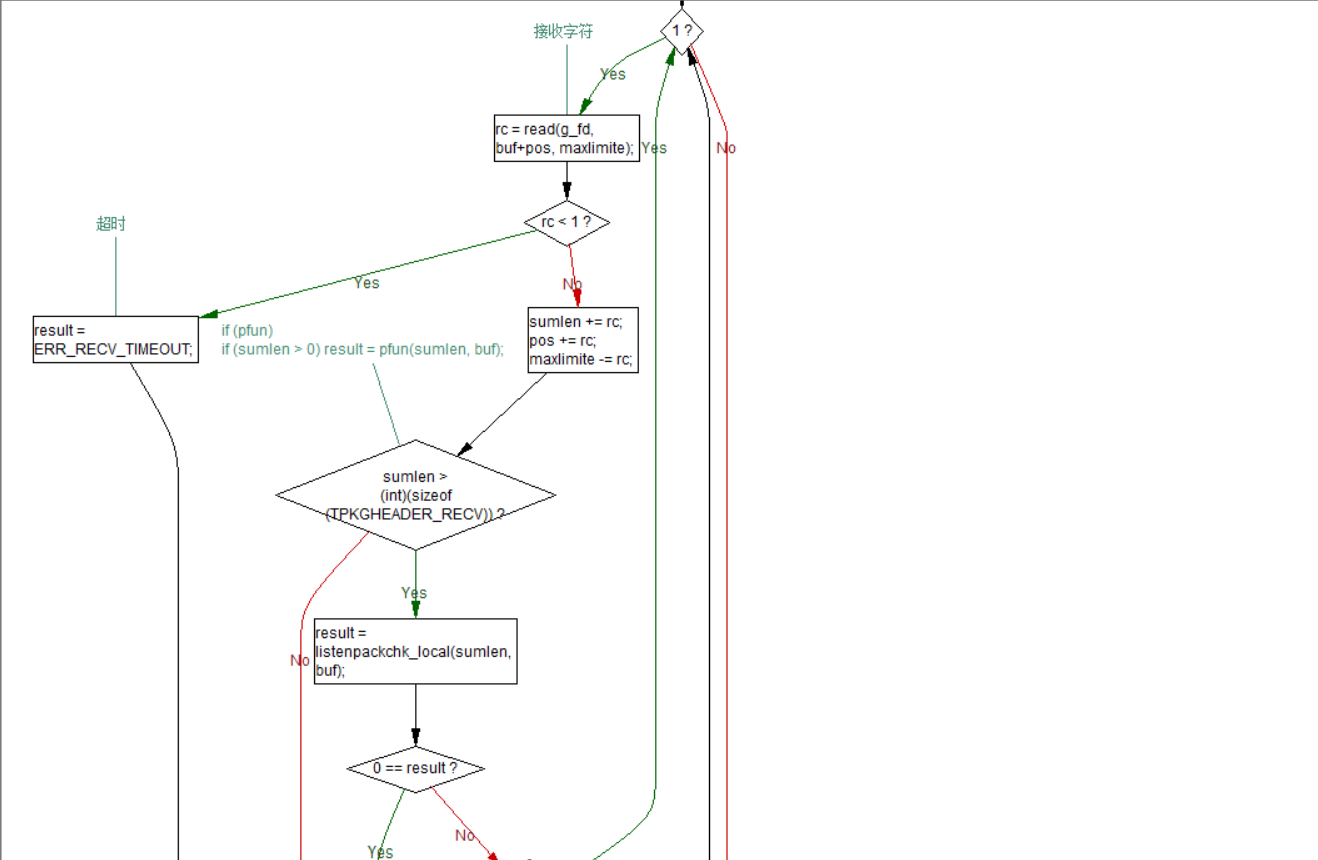
// -3=crc校验错

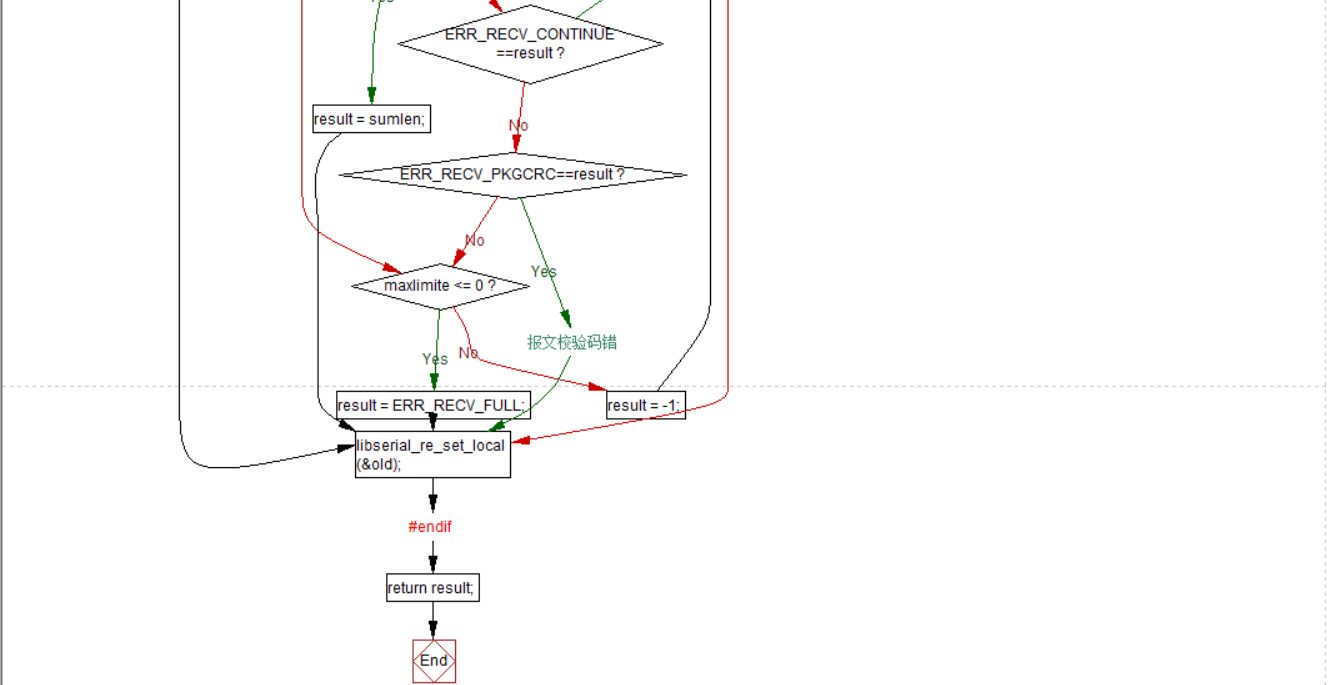
// >0=有正确的数据报文件，返回整个报文长度

//========================================================

int libserial\_recv\_package\_local(const int maxrecv, int timeout, uchar \*buf);







4.2.2 设备初始化

{0, Common\_Initialize\_Device},//设备环境初始化

void Api::Common\_Initialize\_Device(uint8\_t \* param\_stack, uint8\_t \* data\_to\_send, uint16\_t& len\_data)

1、设备初始化函数

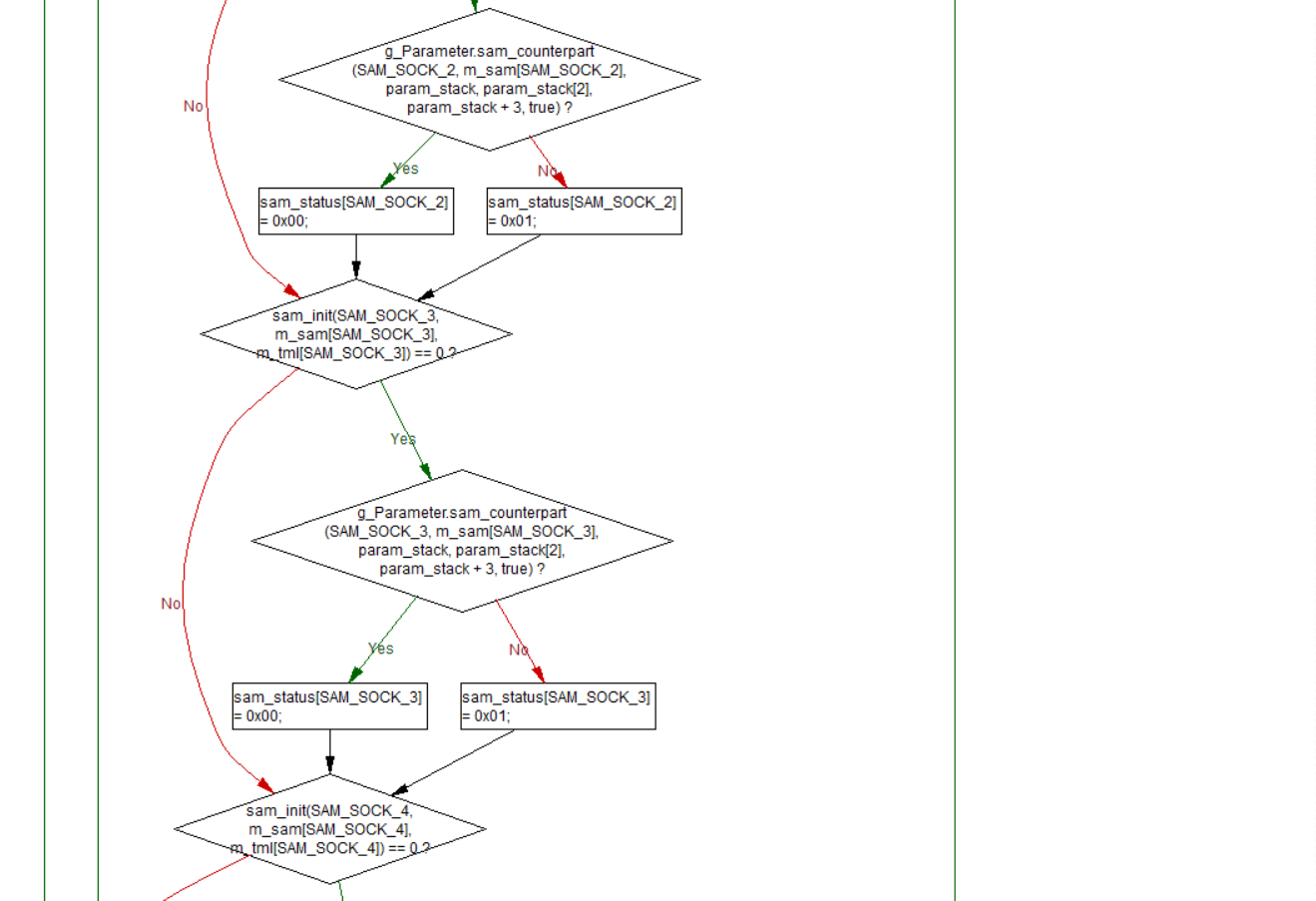
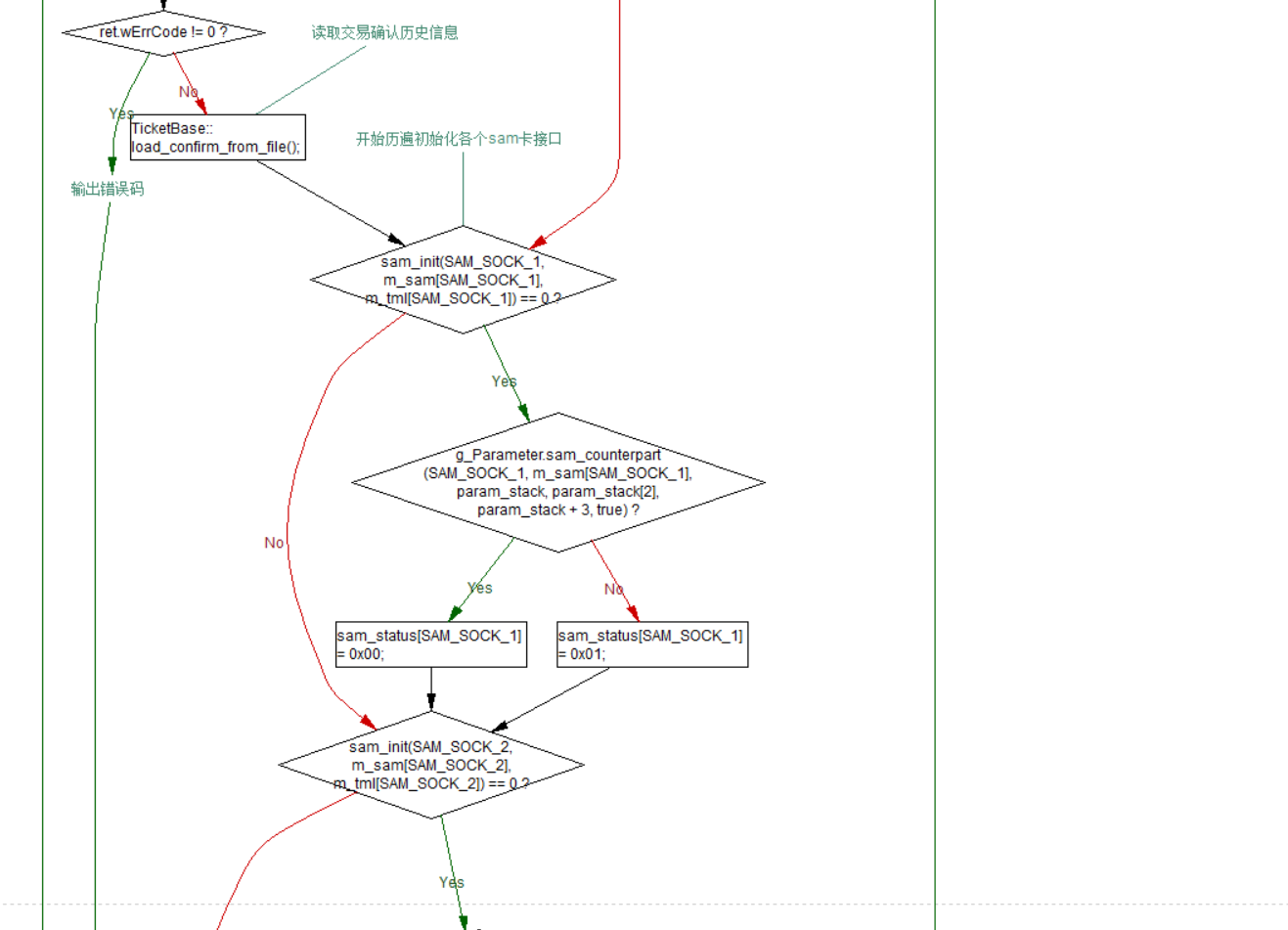
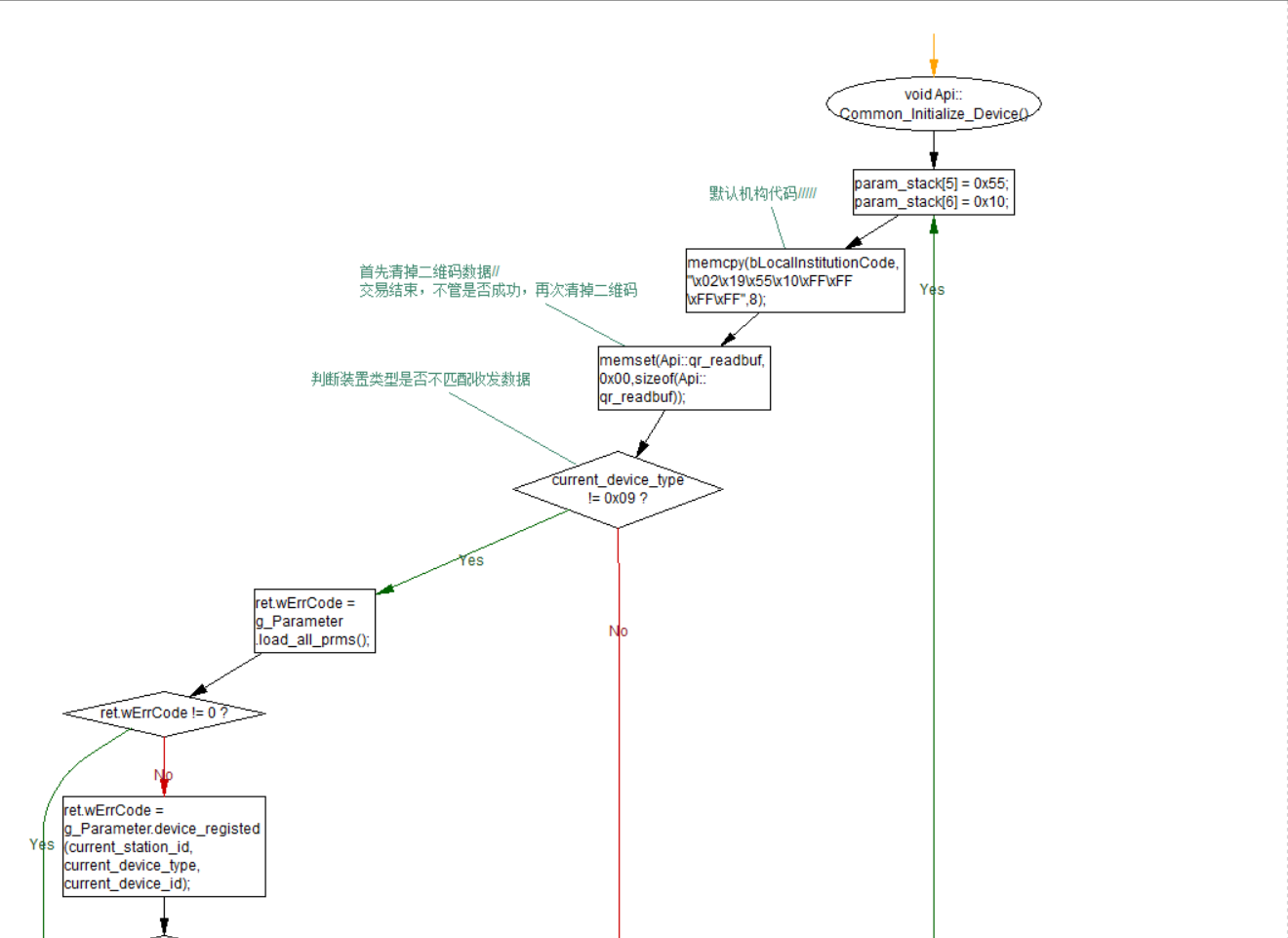
读写器在进行读写操作前必须进行相应的设备初始化操作，只有正确地进行了设备初始化操作后，读写器才能够分别进行TVM入闸出闸处理、BOM型读写器票务处理等，跳过此步骤直接进行票卡处理会使得读写器无法匹配到相对应的票卡信息，使得返回码不为零，出现错误。

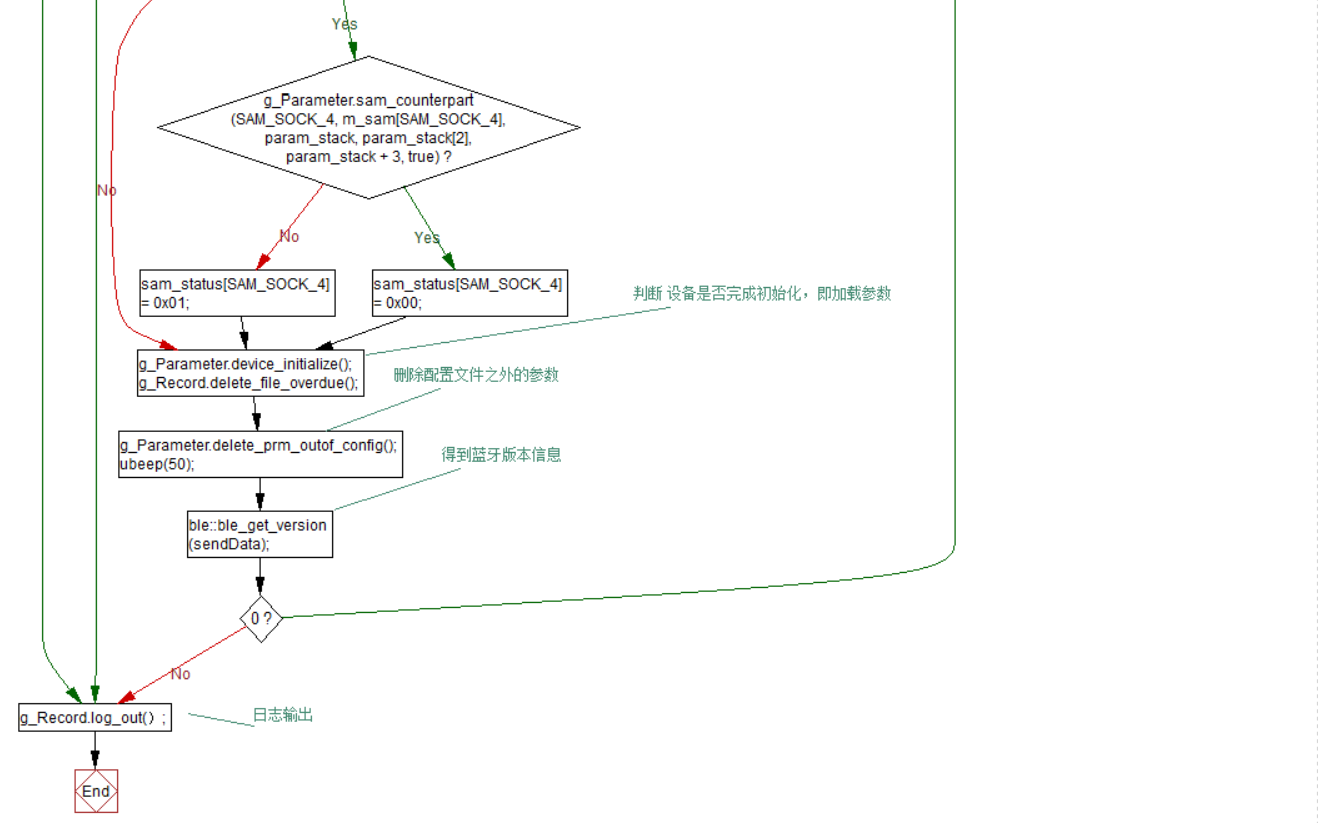
设备环境初始化根据接口传入的参数，加载设备所需参数，申请内部资源。设备要调用读写器处理票卡前必须先执行此调用。设备在初始化之前应先调用‘Config\_Parameter’配置好设备必须的参数，这些参数包括：

|  |  |
| --- | --- |
|  | 参数名称 |
| 1 | 设备控制参数 |
| 2 | 车站代码表 |
| 3 | SAM卡对照表 |
| 4 | 换乘车站代码表 |
| 5 | 票卡参数 |
| 7 | 票价参数 |
| 8 | 降级模式使用记录 |
| 9 | 地铁单个黑名单 |
| 10 | 地铁黑名单段 |
| 11 | 一卡通单个黑名单 |
| 12 | 一卡通黑名单段 |
| 13 | 互联网票单个黑名单 |
| 14 | 互联网票黑名单段 |
| 15 | 行政罚金参数 |
| 16 | 乘次票专用参数 |
| 17 | 闸机专用通道参数 |
| 18 | 二维码扫描器参数 |
| 19 | 蓝牙模块参数 |

其中第17个参数为读写器专用

{0, Common\_Initialize\_Device},//设备环境初始化





4.2.3 版本信息获取

{1, Common\_GetVersion},//获取版本信息

4.2.4 获取SAM卡信息

{2, Common\_GetSamInfo},//获取SAM卡信息

4.2.5 设备降级模式

{3, Common\_SetDegradeMode},//降级模式消息

4.2.6 获取票价

{4, GetFare},//获取到本站的票价

4.2.7 参数管理

{5, Common\_GetParamInfo},//获取参数信息

4.2.8 时间同步

{10, Common\_TimeSynchronization},//时钟同步

4.2.9 获取寄存器值

{6, Get\_Reg\_Value},//获取单个寄存器值

{7, Get\_Reg\_Inf},//获取所有寄存器值

{8, Reset\_Reg\_Inf},//重置寄存器

{9, Common\_LastTrade},//获取上次通讯数据

4.2.10 获取读卡器当前信息

{11, Common\_GetReaderTime},//获取读写器当前时间

**4.3 AGM机业务功能模块详细设计**

4.3.1通道类型处理

1、通道类型函数

{13, Gate\_AisleModel}

void Api::Gate\_AisleModel(uint8\_t \* param\_stack, uint8\_t \* data\_to\_send, uint16\_t& len\_data)

（1）先调用uint16\_t device\_initialized();

判断设备是否完成初始化，即加载参数（读写器异常重启后，设备程序无法知晓，导致没有调用设备初始化）如果无错返回0，有错则返回相应错误码

（2）再调用uint16\_t Api::api\_match\_device(ETPDVC dvc\_type\_support)

传入dvc\_type\_support判断设备是否为AGM机

（3）再次调用uint16\_t Api::api\_match\_device(dvcEntryAGM)

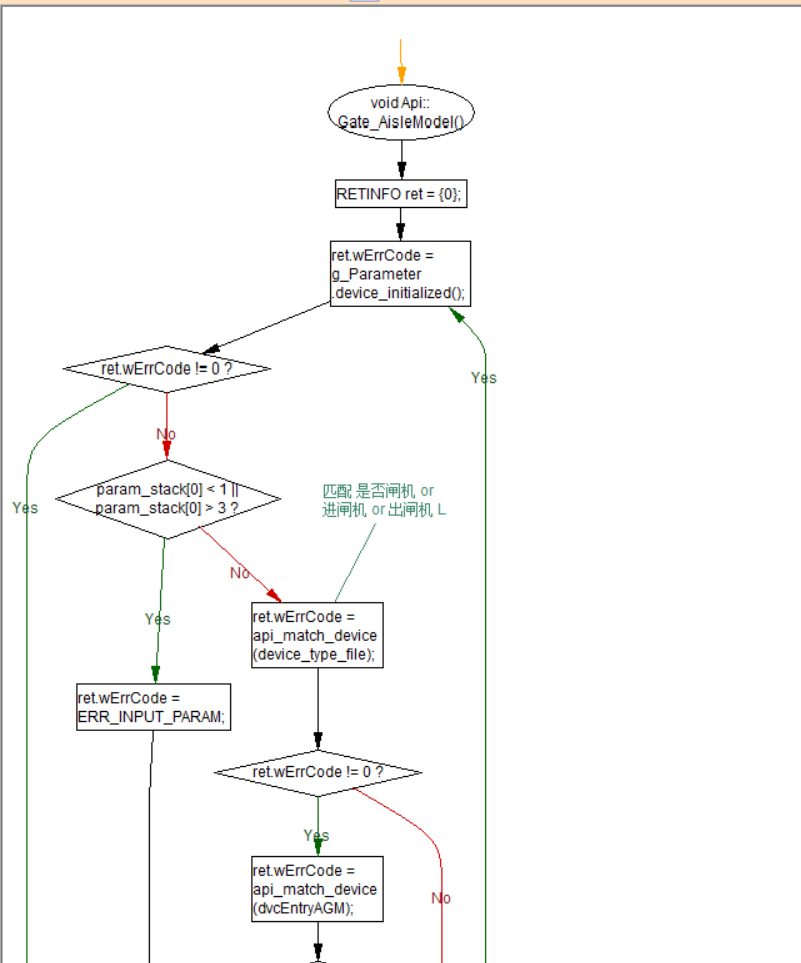
判断是否为入闸机

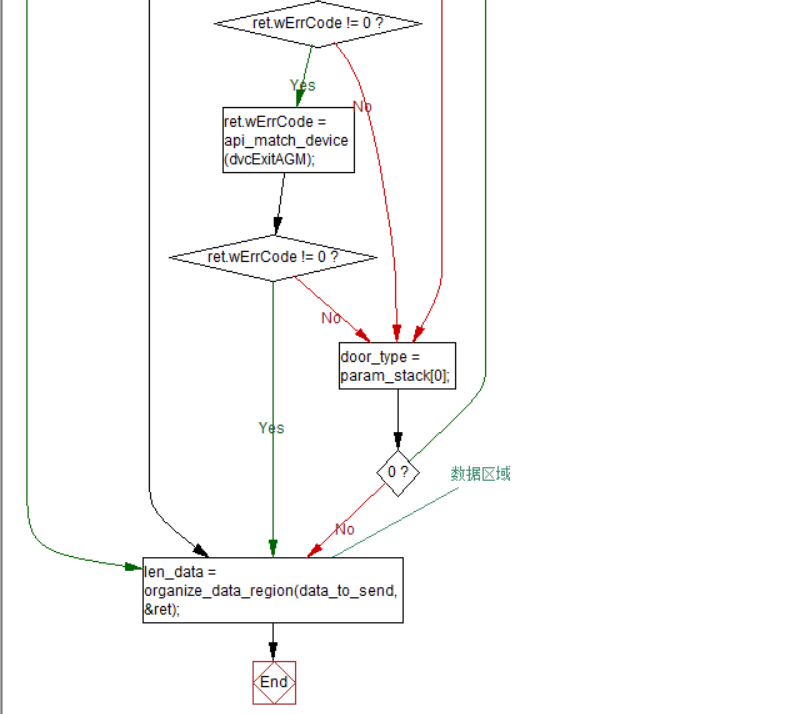
（4）再次调用uint16\_t Api::api\_match\_device(dvcExitAGM)

判断是否为出闸机

（5）最后设定闸机类型door\_type = param\_stack[0];

（6）len\_data = organize\_data\_region(data\_to\_send, &ret);





4.3.2入闸处理

{14, Gate\_EntryFlow},

void Api::Gate\_EntryFlow(uint8\_t \* param\_stack, uint8\_t \* data\_to\_send, uint16\_t& len\_data)

分析票卡开始

ret.wErrCode = g\_Parameter.query\_ticket\_prm(ticket\_data.logical\_type, ticket\_prm);

if (ret.wErrCode) break;

dbg\_formatvar("passageway\_allow\_pass");

检查专用通道

ret.wErrCode = g\_Parameter.passageway\_allow\_pass(ticket\_data.logical\_type, door\_type);

if (ret.wErrCode) break;

dbg\_formatvar("device\_support\_ticket");

设备支持

ret.wErrCode = ParamMgr::device\_support\_ticket(device\_type\_file, ticket\_prm.usable\_device);

if (ret.wErrCode != 0) break;

dbg\_formatvar("analyse\_common");

检查是否重复过闸 如果是电子票卡的话，需要检查重复使用二维吗

if(ticket\_data.qr\_flag == 0x01)

{

if(g\_LoopList.QueryList((char \*)ticket\_data.timeAndcenterCode,operEntry))

{

dbg\_formatvar("this is the same qr code");

ret.wErrCode = ERR\_CARD\_NONE;

memset(Api::qr\_readbuf,0x00,sizeof(Api::qr\_readbuf));

交易结束，不管是否成功，再次清掉二维码

memset(Api::clear\_qr\_readbuf,0x00,sizeof(Api::clear\_qr\_readbuf));//交易结束，不管是否成功，再次清掉二维码

ret = p\_ticket->analyse\_free\_zone(&md\_effect, &ticket\_prm, NULL, NULL);

// 交易确认

ret = p\_ticket->entry\_gate(&entry,&eticketdealinfo);//开门L

if(ret.wErrCode == ERR\_NEED\_CONTINUE\_LAST)//交易确认

memset(Api::qr\_readbuf,0x00,sizeof(Api::qr\_readbuf));//交易结束，不管是否成功，再次清掉二维码

memset(Api::clear\_qr\_readbuf,0x00,sizeof(Api::clear\_qr\_readbuf));//交易结束，不管是否成功，再次清掉二维码

//memset(Api::qr\_readbuf,0x00,sizeof(Api::qr\_readbuf));//交易结束，不管是否成功，再次清掉二维码

4.3.3出闸处理

{15, Gate\_ExitFlow},

void Api::Gate\_ExitFlow(uint8\_t \* param\_stack, uint8\_t \* data\_to\_send, uint16\_t& len\_data)

# 系统测试

## 测试环境

## 功能测试

**5.3 测试结果**

# 总结

。

# 参考文献

# 致谢